The Dissertation Committee for Rashish Tandon
certifies that this is the approved version of the following dissertation:

On Structured and Distributed Learning

Committee:

Alexandros G. Dimakis, Supervisor

Pradeep Ravikumar, Co-Supervisor

Adam Klivans

Eric Price

On Structured and Distributed Learning

by

Rashish Tandon

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2018

Dedicated to my parents.

Acknowledgments

I would like to thank my PhD advisors, Alexandros G. Dimakis and Pradeep Ravikumar, for providing me the opportunity to pursue this work. Both Alex and Pradeep have taught me many valuable lessons on how to be a good researcher and an effective communicator. Their optimism and drive is very infectious, and has helped me to continue pursuing my goals even in moments of setback. Alex has great enthusiasm for research and Pradeep has immense perseverance, and I was lucky enough to have role-models to aspire to for both of those qualities essential for research.

I would also like to thank Adam Klivans and Eric Price for serving on my dissertation committee. I have always enjoyed interacting with them and valued their input. I enjoyed courses taught by them as well: Learning Theory by Adam and Randomized Algorithms by Eric.

Through my PhD, I have had the pleasure of collaborating with multiple people with diverse skills, which has been a transformative learning experience. I want to thank Karthik Shanmugham, Nikos Karampatziakis, Netanel Raviv, Qi Lei, Si Si, Praneeth Netrapalli and Inderjit Dhillon for fruitful collaborations. I also want to thank Sham Kakade and Naveen Goela for mentoring me during my internships.

The importance of friends and lab-mates throughout one's graduate

school life cannot be overstated. I was fortunate enough to have many old and new friends around who served as colleagues, sounding boards and wonderful company. I would like to thank : Nishant, Ankit, Pravesh, Ben, Vivek V., Jahshan, Anand, Abhimanyu, Doug, Prem, Michael, Ashish, Vivek S., Murat, Rajat, Denis, David O., Etienne, Akshay, Matteo, Abhishek, Eshan, Tianyang, David I., Dinesh, to name a few. I am certain however that I have missed many to whom I owe an immense amount of gratitude.

I would also like to thank Lydia Griffith and Katie Dahm for their near instantaneous help with administrative work throughout graduate school.

Last but certainly not the least, I would like to thank my parents and my sister for their continuous love, encouragement and support. It is evident that none of this would have been possible without them.

On Structured and Distributed Learning

Rashish Tandon, Ph.D.
The University of Texas at Austin, 2018

Supervisor: Alexandros G. Dimakis
Co-Supervisor: Pradeep Ravikumar

With the growth in size and complexity of data, methods exploiting low-dimensional structure, as well as distributed methods, have been playing an ever important role in machine learning. These approaches offer a natural choice to alleviate the computational burden, albeit typically at a statistical trade-off. In this thesis, we show that a careful utilization of structure of a problem, or bottlenecks of a distributed system, can also provide a statistical advantage in such settings. We do this from the purview of the following three problems:

- **Learning Graphical models with a few hubs**: Graphical models are a popular tool to represent multivariate distributions. The task of learning a graphical model entails estimating the graph of conditional dependencies between variables. Existing approaches to learn graphical models require a number of samples polynomial in the maximum degree of the true graph, which can be large even if there are a few high-degree

nodes. In this part of the thesis, we propose an estimator that detects and then ignores high degree nodes. Consequently, we show that such an estimator has a lower sample complexity requirement for learning the overall graph when the true graph has a few high-degree nodes or "hubs" for e.g. scale-free graphs.

- **Kernel Ridge Regression via partitioning**: Kernel methods find wide and varied applicability in machine learning. However, solving the Kernel Ridge Regression (KRR) optimization requires computation that is cubic in the number of samples. In this work, we consider a divide-and-conquer approach to solve the KRR problem. The division step involves splitting the samples based on a partitioning of the input space, and the conquering step is to simply use the local KRR estimate in each partition. We show that this can not only lower the computational requirements of solving the KRR problem, but also lead to improved accuracy over both a single KRR estimate, and estimates based on random data partitioning.

- **Stragglers in Distributed Synchronous Gradient Descent**: Synchronous methods in machine learning have many desirable properties, but they are only as fast as the slowest machine in a distributed system. The straggler/slow machine problem is a critical bottleneck for such methods. In this part of our work, we propose a novel framework based on Coding Theory for mitigating stragglers in Distributed Synchronous Gradient Descent (and its variants). Our approach views

stragglers as errors/erasures. By carefully replicating data blocks and coding across gradients, we show how this can provide tolerance to failures and stragglers without incurring any communication overheads.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the current big data era, data sets have been growing at a humongous rate, both in terms of the number of samples, n, as well as the dimensionality of each sample, p. This rapid growth has prompted a renewed interest in machine learning approaches that exploit a low-dimensional structure in the problem, as well as distributed algorithms for machine learning. While the ambient dimension of a problem may be high, having low-dimensional structure, such as sparsity, low-rank etc., makes the problem amenable to efficient estimators. Alternatively, distributed versions of machine learning algorithms can help alleviate the computational burden for an estimator. Both these paradigms are accompanied with their own set of assumptions, sample complexity requirements, statistical guarantees and bottlenecks — and using these as black-boxes for many real-world settings may not be the best thing to do.

In this thesis, we study three problems of interest in machine learning — Learning Graphical models, Kernel Ridge Regression and Distributed Synchronous Gradient Descent. We show that further consideration for structure or bottlenecks in these problems, whilst in a distributed setting, can lead to an improvement in statistical performance over several estimators. Specifically,

we consider the following problems:

- **Learning graphical models with a few hubs**: A graphical model is a useful tool for representing multivariate distributions. It comprises of a Markov Graph which represents various conditional dependencies in a distribution as edges in a graph. The task of learning a graphical model entails estimating this Markov graph, given samples from the underlying distribution. A common approach to do this is based on estimating the neighborhood of each node and then combine them to obtain a global estimate. Learning the neighborhood of any node typically requires a number of samples polynomial in the degree of the node. This can be a problem if there are some high-degree nodes in the true underlying graph, since estimating them may require way more samples than what we have. To avoid this, we propose an estimator which uses fewer samples by detecting and then ignoring high degree nodes. Thus, we only require learning the neighborhoods of low degree nodes. We do this for the problem of learning Ising models (a subclass of graphical models). Consequently, we show that such an estimator has a lower sample complexity requirement for learning the overall graph when the true graph has a few high-degree nodes or "hubs" for e.g. scale-free graphs.

- **Kernel Ridge Regression (KRR)**: Kernel methods are widely used in machine learning, since they provide a simple mechanism to extend many linear models to more complex functions. By using the Kernel trick and

the Representer theorem, they allow learning linear models implicitly in a space of higher-dimension (potentially infinite) while still keeping the optimization tractable. They suffer, however, from high computational requirements: typically polynomial in the number of samples. For example, solving a single KRR optimization has a computational complexity that is cubic in the number of samples. A simple and commonly used distributed strategy to reduce the computational cost is to randomly split the data into disjoint groups, learn a KRR estimate for each group, and then return the average of all estimates. By controlling the number of groups, one can tradeoff the overhead of learning multiple KRR estimates with the gain in computation of any of the individual KRR estimates. Note that the latter is lesser as groups increase since each group has fewer samples. In our work, we consider an alternate divide-and-conquer approach to solve the KRR problem. The division step involves splitting the samples based on a partitioning of the input space (obtained via clustering, or otherwise), and the conquering step is to simply use the local KRR estimate of a partition (i.e. a KRR estimate using points only in the partition) as its global estimate. We show that this can not only lower the computational requirements of solving the KRR problem, but also lead to a statistical improvement over both a single KRR estimator and estimators based on random data splitting alluded to above.

- **Stragglers in Distributed Synchronous Gradient Descent**: Syn-

chronous methods in distributed machine learning have many desirable properties such as stability and faster convergence. A major drawback however is that they are only as fast as the slowest machine in a distributed system. In this work, we propose a novel approach to tackling this problem. We leverage ideas from coding theory to mitigate stragglers in Distributed Synchronous Gradient Descent. Typically, coding theory is used for reliable communication across a channel susceptible to errors/erasures. By carefully encoding the intended message as a larger message, one can guarantee recovery of the original message up to a certain number of errors. Our key idea is to view stragglers as errors/erasures from a coding theoretic perspective. For our setting, we show that by carefully replicating data blocks and coding across the gradients (in in Distributed Synchronous Gradient Descent), one can obtain tolerance to failures and stragglers at no communication overhead, although with a computational one.

The rest of this document is organized as follows. Chapter 2 discusses the problem of Learning Graphs with a few hubs. Chapter 3 relates to the problem of Kernel Ridge Regression. Chapter 4 considers the problem of stragglers in Distributed Synchronous Gradient Descent. Finally, Chapter 5 concludes with a short summary.

Chapter 2

Learning Graphs with a Few Hubs[1]

2.1 Introduction

Graphical Models are a popular class of multivariate probability distributions that are widely used in applications across science and engineering. The key idea here is to represent probability distributions compactly as a product of functions over the cliques of an underlying graph. The task of graphical model selection is to learn the underlying undirected graph given samples drawn from the distribution it represents. This task becomes particularly difficult in high-dimensional data settings, where the number of variables p could be larger than the number of samples n.

Due in part to its importance, many practical algorithms with strong statistical guarantees have been proposed for this graphical model selection problem. In this chapter, we focus on binary Ising models, *i.e.* where the variables are binary. For such Ising graphical models, [49] show that "local" node-wise ℓ_1-regularized logistic regressions can recover the underlying graph

[1]This chapter is based on [60]. The author of this work was the first author and primary contibutor to [60]. The author proposed the problem as well as the solution considered here, analyzed the performance of the solution theoretically, and performed experiments to validate the analysis empirically. The author also collaborated on writing the paper.

exactly with high probability, when given $n = O(d^3 \log(p))$ i.i.d. samples, where p is the number of nodes, and d is the maximum node-degree of the graph. Another class of methods are based on local search and thresholding [1, 3, 9, 13], but in the absence of other stringent assumptions, their computational complexity scales exponentially with the local node-degrees d. Among more "global" approaches, [13, 30, 43] and others have proposed penalized pseudo-likelihood [7] based approaches; while [61] have proposed penalized estimators based on variational approximations to the graphical model log-likelihood; however the sample complexity of these methods also scale polynomially with the maximum node-degree of the graph.

Here, we consider the setting where the graphs have a few *hub nodes*, which are highly connected nodes whose degree could scale as large as linearly in the number of nodes. An importance instance of this are *power-law graphs*, which occur ubiquitously in many real-world settings, and in which hub-nodes with large degrees are few in number but not non-existent, and their maximum node degree could be very large. Since the sample complexity of the state of the art methods listed above scale polynomially with the maximum node-degree, they would thus not be very suitable in recovering such power-law graphs with hub nodes. Motivated by this, there have been a few statistical estimators proposed that explicitly target power-law graphical model estimation. [38] propose a novel *non-convex* regularization motivated by the power-law degree distribution, a convex variant of which was also considered in [19]. While these methods did not provide theoretical guarantees, even their experimen-

tal results demonstrated limited improvements in sample complexity over ℓ_1 regularization based methods. [43] propose a pseudo-likelihood based procedure for learning discrete graphical models that minimizes the sum of *weighted* node-wise conditional log-likelihoods, where the node-wise weights could potentially be tuned to encourage power-law structure, but this was suggested as a heuristic. For the specific case of Gaussian graphical models, [26] provide an approach based on thresholding sample partial correlation matrices, and provide asymptotic expressions for false discovery rates under stringent weak dependence assumptions.

Consider the following leading question: what if we do not have enough samples to solve for the node-conditional distribution of a hub-node in an Ising model i.e. what if we we have less than $d_h^3 \log p$ samples, where d_h is the degree of the hub node? The estimators above that focus on the estimation of a hub-networked graphical model all focus in part on the estimation of such "difficult" sub-problems; so that they have a large sample complexity for estimating such hub-networked graphical models [58]. Instead, we propose to turn the problem on its head, and use our *inability* to estimate such difficult sub-problems given limited samples, to then turn around and be able to estimate the hub-network. To provide intuition for our strategy, consider a star-shaped graph, with one hub node, and the rest being spoke nodes connected only to the hub. The maximum degree of the hub node is thus $p-1$, so that estimating the node-conditional distribution of the hub-node would require samples scaling as $p^3 \log p$. What if only have samples scaling as $\log p$? But suppose

we are also able to realize that we are *unable* to estimate the node-conditional distribution of the hub-node; and only those of the spoke nodes. We can then ignore the neighborhood estimation of the hub-node, and use the reliable neighborhood estimates of just the spoke nodes: this suffices to estimate the star-graph.

In this work, we formalize this strategy: we provide a quantitative criterion for checking whether or not the given number of samples suffice for regularized node-conditional distribution estimation as in [49] at a given node. We then use this to detect "hub nodes," and use only the neighborhood estimates from the remaining nodes to construct the graph estimate. We note that our notion of "hub nodes" is specifically related to the difficulty of node-neighborhood estimation, which only roughly corresponds to the node-degree (while the required sample size scales as $O(d^3 \log p)$, the constants matter in finite sample settings).

Our criterion is based on the following key observations on ℓ_1 regularized node-neighborhood estimation for any node $u \in V$ conditioned on the rest of the nodes. Consider the variance of the Bernoulli event of the incidence of any node $v \in V \backslash u$ in the node-neighborhood estimate, as a function the regularization penalty. When the penalty is very small, the node-neigborhood estimate will include all nodes, and the variance will be zero; when the penalty is "just right," the node-neighborhood estimate will be correct and will include v iff it is a neighbor with very high probability, so that the variance will again be (close to) zero, and when the penalty is very large, the node-neighborhood

estimate will be null, and the variance will again be zero. Contrast this behavior with the setting where there are very few samples to allow for neighborhood recovery at any value of the regularization penalty: then the variance starts off at zero, rises, and then slowly goes to zero as the node-neighborhood becomes null. The difference in the observable behaviors between these two settings thus allows us to differentiate "hub" nodes from non-hubs. As we show, we are able to provide concrete statistical guarantees for our procedure, demonstrating improved sample complexity over the vanilla ℓ_1 regularized node-regression procedure.

We note that the approach of [39] is similar in spirit to ours, utilizing a weighted combination of the node wise estimates to obtain the overall estimate, where the weights are the inverse of an alternate notion of variance. However, their approach deals with parameter estimation in the asymptotic sense, and is not applicable to structure estimation in the high dimensional setting.

Overall, we make a key advance in the estimation of hub-networked graphical models: we provide a tractable procedure with strong statistical guarantees even under very low-sample settings where we cannot even estimate the node-conditional distributions of the hub nodes. Our methods involve binary reliability indicators for node-conditional distribution estimation, which could have broader applications in many scientific and engineering applications, even outside the context of graphical model estimation. Finally, all proofs related to the results in this Chapter can be found in Appendix A.

2.2 Notation and Preliminaries

Let $X = (X_1, \ldots, X_p)$ be a random vector, with each variable $X_i\,(i \in [p])$ taking values from a discrete set $\mathcal{X}$. Let $G = (V, E)$ be an undirected graph over p nodes, corresponding to the p variables $\{X_1, \ldots, X_p\}$. A pairwise Markov random field over $X = (X_1, \ldots, X_p)$ is a probability distribution specified by non-negative pairwise functions $\phi_{rt} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ for each edge $(r, t) \in E$:

$$\mathbb{P}(x) \propto \prod_{rt \in E} \phi_{rt}(x_r, x_t) \tag{2.1}$$

Note that we use rt as a shorthand for the edge (r, t). In this work, we focus on the Ising model setting *i.e.* where we have binary variables with $\mathcal{X} = \{-1, 1\}$, and where $\phi_{rt} = \exp(\theta_{rt} x_r x_t)$ for a given set of parameters $\theta = \{\theta_{rt} \,|\, rt \in E\}$. In this case, (2.1) can be rewritten as :

$$\mathbb{P}_\theta(x) = \frac{1}{Z(\theta)} \exp\left\{\sum_{rt \in E} \theta_{rt} x_r x_t\right\}, \tag{2.2}$$

where $Z(\theta) = \sum\limits_{x \in \{-1,1\}^p} \exp\left\{\sum_{rt \in E} \theta_{rt} x_r x_t\right\}$.

Let $D := \{x^{(1)} \ldots, x^{(n)}\}$ be n samples drawn i.i.d from the Ising model distribution P_{θ^*} with parameters $\theta^* \in \mathbb{R}^{\binom{p}{2}}$ and Markov graph $G^* = (V, E^*)$, $|V| = p$. Note that each sample $x^{(i)}$ is a p-dimensional binary vector $x^{(i)} \in \{-1, 1\}^p$. The edge set E^* is related to the parameters θ^* as $E^* = \{(r, t) \in V \times V \,|\, \theta^*_{rt} \neq 0\}$.

The task of graphical model selection is to infer this edge set E^* using the n samples. Any estimator $\widehat{E}_n$ for this task is said to be sparsistent if it satisfies $\mathbb{P}\left[\widehat{E}_n = E^*\right] \to 1$ as $n \to \infty$.

2.2.1 ℓ_1-regularized estimator

We now briefly review the state-of-the-art estimator of [49] (called the ℓ_1-estimator henceforth). The key idea there is to estimate the true graph E^* by estimating the neighbourhood of each node $r \in V$ in turn. Suppose $\mathcal{N}^*(r)$ denotes the true neighbours of the vertex r, so that $\mathcal{N}^*(r) = \{t \,|\, (r,t) \in E^*\}$. The ℓ_1-estimator uses sparsistent neighborhood estimators $\widehat{\mathcal{N}}_n(r) \subset V \; \forall \, r \in V$ s.t. $\mathbb{P}\left[\widehat{\mathcal{N}}_n(r) = \mathcal{N}^*(r)\right] \to 1$ as $n \to \infty$, to then obtain a sparsistent estimate of the entire graph.

Note that for any $r \in V$, the set of parameters θ^* is related to the true neighbourhood as $\mathcal{N}^*(r) = \{t \,|\, \theta^*_{rt} \neq 0, t \in V\}$. The ℓ_1-estimator exploits this to pose neighbourhood selection as an ℓ_1-regularized logistic regression problem, minimizing the negative conditional log-likelihood for each node with an additional ℓ_1-penalty. Note that for a set of parameters θ and a node $r \in V$, the conditional distribution of X_r conditioned on $X_{V\setminus r}$ is given as

$$\mathbb{P}_\theta\left(x_r \,|\, x_{V\setminus r}\right) = \frac{\exp(2x_r \sum_{t\in V\setminus r} \theta_{rt} x_t)}{1 + \exp(2x_r \sum_{t\in V\setminus r} \theta_{rt} x_t)}. \tag{2.3}$$

Defining $\theta_{\setminus r} = \{\theta_{rt} \,|\, t \in V, t \neq r\}$ and $x_{\setminus r} = \{x_t \,|\, t \in V, t \neq r\}$, the negative conditional log-likelihood of the samples D would be given by

$$\begin{aligned} &\mathcal{L}(\theta_{\setminus r}; D) = \\ &\frac{1}{n}\sum_{i=1}^{n} \left\{ \log\left(1 + \exp\left(2x_r^{(i)} \theta_{\setminus r}^T x_{\setminus r}^{(i)}\right)\right) - 2x_r^{(i)} \theta_{\setminus r}^T x_{\setminus r}^{(i)} \right\}. \end{aligned} \tag{2.4}$$

The ℓ_1-estimator solves the following optimization problem for each $r \in V$:

$$\underset{\theta_{\setminus r} \in \mathbb{R}^{p-1}}{\arg\min} \left\{ \mathcal{L}(\theta_{\setminus r}; D) + \lambda \|\theta_{\setminus r}\|_1 \right\}. \tag{2.5}$$

Let $\widehat{\theta}_{\backslash r}(D)$ correspond to the solution of (2.5). Then the neighbourhood estimate is given as the non-zero locations or support of $\widehat{\theta}_{\backslash r}(D)$: $\widehat{\mathcal{N}}_\lambda(r;D) = \text{Support}\left(\widehat{\theta}_{\backslash r}(D)\right)$. Finally, the edge estimate is computed by taking the union of all neighbourhood estimates: $\widehat{E}_{n,\lambda} = \underset{r\in V}{\cup}\{(r,t)\,|\,t\in\widehat{\mathcal{N}}_\lambda(r;D)\}$.

The ℓ_1-estimator has been shown to have strong statistical guarantees under certain incoherence conditions. Below, we restate the incoherence conditions of [49], for the sake of completeness. These are stated in terms of the Hessian (in expectation) of the likelihood function for the true parameter vector $\theta^*_{\backslash r}$, which is given as $Q^*_r = \mathbb{E}\left[\nabla^2 \log P_{\theta^*}\left(x_r\,|\,x_{V\backslash r}\right)\right]$. For brevity, we shall briefly write Q^*_r as Q^*, the true neighbourhood set $\mathcal{N}^*(r)$ as $\mathcal{N}$, and its complement, $V\setminus\mathcal{N}^*(r)$ as $\mathcal{N}^c$. Then, their incoherence conditions (with $r\in V$ being implicit in Q^* and $\mathcal{N}$) are :

(A1) $\exists$ a const. $C_{\min} > 0$ s.t. $\Lambda_{\min}\left(Q^*_{\mathcal{N}\mathcal{N}}\right) \geq C_{\min}$. Also, $\exists$ a const. $C_{\max}$ s.t. $\Lambda_{\max}\left(\mathbb{E}\left[X_{V\backslash r}\,X^T_{V\backslash r}\right]\right) \leq C_{\max}$

(A2) $\exists$ a constant $\alpha\in(0,1]$ s.t. $\left\|Q^*_{\mathcal{N}^c\mathcal{N}}\left(Q^*_{\mathcal{N}\mathcal{N}}\right)^{-1}\right\|_\infty \leq 1-\alpha$

Note that $\Lambda_{\min}(\cdot)$ and $\Lambda_{\max}(\cdot)$ correspond to the minimum and maximum eigenvalues of a matrix respectively, and $\|\cdot\|_\infty$ corresponds to the standard ℓ_∞-matrix norm.

Now, we restate the main theorem below from [49] using our notation, and refer the reader to their paper for details.

Theorem 2.1 (Guarantee for the ℓ_1-estimator; see [49]). *Suppose an Ising graphical model with true parameter set θ^* satisfies conditions $(A1)$ and $(A2)$ for all nodes $r \in V$. Consider any $r \in V$, and let $d_r = \|\theta^*_{\backslash r}\|_0$ denote its degree. Then, there exist constants c_1, c_2, c_3, c_4 such that if we have $\lambda \geq c_1\sqrt{\frac{\log p}{n}}$ and $n > c_2\, d_r^3 \log p$ and $\mathcal{N}^*_{sub}(r) = \left\{ t \in \mathcal{N}^*(r) \,\middle|\, |\theta^*_{rt}| \geq c_3\sqrt{d_r}\lambda \right\}$, then*

$$\mathbb{P}\left(\widehat{\mathcal{N}}_\lambda(r; D) = \mathcal{N}^*_{sub}(r)\right) \geq 1 - 2\exp\left(-c_4\lambda^2 n\right). \tag{2.6}$$

Based on Theorem 2.1, and a simple application of the union bound, we can see that the sample complexity for recovering the entire graph scales as $n = \Omega(d^3_{\max} \log p)$ samples, where $d_{\max}$ is the maximum degree of the graph $G^* = (V, E^*)$. However, as detailed earlier, $d_{\max}$ may be huge for hub-graphs, so that the sample complexity of the ℓ_1-estimator will be large for such graphs.

2.3 Sufficiency Measure Based Estimator

As noted in the introduction, our approach is based on using a quantitative criterion for checking whether or not the given number of samples suffice for regularized node-conditional distribution estimation as in the ℓ_1-estimator at a given node. Given such a criterion, we can then take the union of only those neighborhood estimates which the method is guaranteed to estimate accurately, and not consider the "junk" estimates. Towards building such a observable "sufficiency" criterion, we first setup some notation.

2.3.1 Sufficiency Measure

For every $r \in V$ and $t \in V \setminus r$, we define $p_{r,n,\lambda}(t) = \mathbb{P}\left(t \in \widehat{\mathcal{N}}_\lambda(r; D)\right)$, as the probability of variable t being included in the neighbourhood estimate of variable r, estimated by the ℓ_1-estimator with regularization λ, given n samples drawn i.i.d. from the underlying Ising model. Note that the probability is taken over n samples. Based on Theorem 2.1, we have the following simple corollary.

Corollary 2.1. *For any $r \in V$, suppose θ^* and (n, λ) satisfy all conditions of Theorem 2.1 with constants c_1, c_2, c_3, c_4; then*

$$\begin{aligned} p_{r,n,\lambda}(t) &\geq 1 - 2\exp\left(-c_4\lambda^2 n\right) \qquad \text{if } t \in \mathcal{N}^*_{sub}(r) \text{ and,} \\ p_{r,n,\lambda}(t) &\leq 2\exp\left(-c_4\lambda^2 n\right) \qquad \text{if } t \notin \mathcal{N}^*_{sub}(r), \end{aligned} \tag{2.7}$$

where $\mathcal{N}^*_{sub}(r) = \left\{ t \in \mathcal{N}^*(r) \,\middle|\, |\theta^*_{rt}| \geq c_3\sqrt{d}\lambda \right\}$.

Thus, when the number of samples n is sufficient for neighborhood recovery, depending on whether node t is in the true neighborhood of r, $p_{r,n,\lambda}(t)$ goes extremely close to zero or one; equivalently $p_{r,n,\lambda}(t)\left(1 - p_{r,n,\lambda}(t)\right)$ goes extremely close to zero. Building on this observation, let us define the "sufficiency" measure

$$\mathcal{M}_{r,n,\lambda} = \max_{t \in V \setminus r} \; p_{r,n,\lambda}(t)\left(1 - p_{r,n,\lambda}(t)\right). \tag{2.8}$$

It can thus be seen that this sufficiency measure goes to zero when the number of samples n is sufficient for recovering the neighborhood of node r.

In the sequel, we will analyze a natural U-statistic to estimate this sufficiency measure from data. We first require some more notation. For any b $\left(1 < b < \frac{n}{2}\right)$, we define $S_b(D)$ as the set of all possible subsamples of size b, drawn from D without replacement, so that

$$S_b(D) = \{(x^{(i_1)}, \ldots, x^{(i_b)}) \,|\, 1 \leq i_1 < \ldots < i_b \leq n\}. \tag{2.9}$$

Given any subsample $D_b \in S_b(D)$ of size b, let $F^t_{\lambda,r}(D_b)$ be a function such that

$$F^t_{\lambda,r}(D_b) = \begin{cases} 1 & \text{if } t \in \widehat{\mathcal{N}}_{b,\lambda}(r; D_b) \\ 0 & \text{otherwise.} \end{cases} \tag{2.10}$$

Now, we consider the U-statistic (of order b),

$$\widetilde{p}_{r,b,\lambda}(t; D) = \frac{1}{\binom{n}{b}} \sum_{D_b \in S_b(D)} F^t_{\lambda,r}(D_b). \tag{2.11}$$

Note that $\mathbb{E}\left[\widetilde{p}_{r,b,\lambda}(t; D)\right] = p_{r,b,\lambda}(t)$. We are now ready to provide the U-statistic estimate of the sufficiency measure in (2.8):

$$\widetilde{\mathcal{M}}_{r,b,\lambda}(D) = \max_{t \in V \setminus r} \widetilde{p}_{r,b,\lambda}(t; D) \left(1 - \widetilde{p}_{r,b,\lambda}(t; D)\right). \tag{2.12}$$

Computing $\widetilde{\mathcal{M}}_{r,b,\lambda}(D)$ would require computing $\widetilde{p}_{r,b,\lambda}(t; D)$ for every $t \in V \setminus r$, which in turn would require considering all possible $\binom{n}{b}$ sub-samples of D. However, as we show below (see also analyses in [37, 44] on sub-sampling), it suffices to choose a number $N \geq n/b$ of subsamples drawn at random. Thus, our actual estimate for $p_{r,b,\lambda}(t)$ is

$$\hat{p}_{r,b,\lambda}(t; D) = \frac{1}{N} \sum_{i=1}^{N} F^t_{\lambda,r}(D_i), \tag{2.13}$$

where $D_1, \ldots, D_N$ are subsamples chosen independently and uniformly at random from $S_b(D)$, and the estimate for the sufficiency measure is

$$\widehat{\mathcal{M}}_{r,b,\lambda}(D) = \max_{t \in V \setminus r} \hat{p}_{r,b,\lambda}(t; D) \left(1 - \hat{p}_{r,b,\lambda}(t; D)\right). \tag{2.14}$$

We describe the procedure to calculate $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$ in Algorithm 2.1.

Algorithm 2.1 Estimating $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$

Input : Data $D := \{x^{(1)}, \ldots, x^{(n)}\}$, Regularization parameter λ, Sub-sample size b , No. of sub-samples N

Output: An estimate of $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$

$\forall\, t \in V \setminus r, \quad \hat{p}_{r,b,\lambda}(t; D) \leftarrow 0$

for $i = 1$ **to** N **do**

  Pick a sub-sample D_i chosen uniformly randomly from $S_b(D)$

  Compute $\widehat{\mathcal{N}}_{b,\lambda}(r; D_i)$ by solving (2.5) (ℓ_1-estimate)

  for $t \in \widehat{\mathcal{N}}_{b,\lambda}(r; D_i)$ **do**

    $\hat{p}_{r,b,\lambda}(t; D) \leftarrow \hat{p}_{r,b,\lambda}(t; D) + 1$

$\forall\, t \in V \setminus r, \quad \hat{p}_{r,b,\lambda}(t; D) \leftarrow \hat{p}_{r,b,\lambda}(t; D)/N$

$\widehat{\mathcal{M}}_{r,b,\lambda}(D) \leftarrow \max_{t \in V \setminus r} \hat{p}_{r,b,\lambda}(t; D) \left(1 - \hat{p}_{r,b,\lambda}(t; D)\right)$

Once $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$ has been computed, we have the following lemma which shows that it is ϵ-close to $\mathcal{M}_{r,b,\lambda}$ with high probability, provided we have sufficiently many samples .

Proposition 2.1 (Concentration of $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$ to $\mathcal{M}_{r,b,\lambda}$)**.** *For any $\delta \in (0, 1]$ and $\epsilon > 0$, if we have $n > \frac{18b}{\epsilon^2}\left[\log p + \log\left(4/\delta\right)\right]$ and $N \geq \lceil \frac{n}{b} \rceil$, then,*

$$\mathbb{P}\left(|\widehat{\mathcal{M}}_{r,b,\lambda}(D) - \mathcal{M}_{r,b,\lambda}| \leq \epsilon\right) \geq 1 - \delta. \tag{2.15}$$

2.3.2 Behavior of the Sufficiency Measure

The key question of interest is whether we can use the sufficiency measure $\mathcal{M}_{r,b,\lambda}$ (via its sample estimate $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$) to detect "hub-nodes" that we define specifically as those nodes for which we do not have enough samples for the ℓ_1-estimator to be sparsistent. Correspondingly, let us define "non-hub" nodes in this context as those nodes for which we do have enough samples for the ℓ_1-estimator to be sparsistent. We formalize these notions below.

Definition 1 (Non-Hub Node vs. Hub Node)**.** *Assume that the true parameter set θ^* satisfies the incoherence conditions, $(A1)$ and $(A2)$, for all nodes $r \in V$. Consider any node $r \in V$. It is termed a "non-hub node" w.r.t. n samples if $\exists$ a regularization parameter λ s.t. (n, λ) satisfy all conditions of Theorem 2.1 with constants c_1, c_2, c_3, c_4. Otherwise, the node is termed a "hub" node.*

Since the sample complexity of neighborhood estimation via the ℓ_1-estimator scales cubically with the node-degree (from Theorem 2.1), hub nodes as we define here correspond loosely to high-degree nodes, but in the sequel, the exact specification of "hub" and "non-hub" nodes are as detailed by the definition above.

Before we describe the behaviour of $\mathcal{M}_{r,b,\lambda}$ for "hub" nodes and "non-hub" nodes, we impose the following technical assumptions on the behaviour of $p_{r,b,\lambda}(t)$ needed for our algorithm to work.

Assumption 2.1. *$\forall\, r \in V$, $p_{r,b,\lambda}(t)$ satisfies the following: For fixed b and*

some constant $c(> 0)$*, let*

$$\lambda_{\min}(t) = \min\left\{\lambda \geq 0 \,|\, p_{r,b,\lambda}(t) \leq 1 - 2\exp\left(-c\log p\right)\right\},$$

$$\textit{and,} \tag{2.16}$$

$$\lambda_{\max}(t) = \max\left\{\lambda \geq 0 \,|\, p_{r,b,\lambda}(t) \geq 2\exp\left(-c\log p\right)\right\}.$$

Then, $\lambda_{\min}(t)$ *and* $\lambda_{\max}(t)$ *are attained at finite values s.t.*

(a) For any $t \in V \setminus r$ *and* $\lambda \in (\lambda_{\min}(t), \lambda_{\max}(t))$*, we have*

$$p_{r,b,\lambda}(t) \in [2\exp(-c\log p), 1 - 2\exp(-c\log p)]. \tag{2.17}$$

(b) For all $t \notin \mathcal{N}^*(r)$*,*

$$\lambda_{\min}(t) \leq \lambda_{\min} < \lambda_{\max} \leq \lambda_{\max}(t), \tag{2.18}$$

for some finite $\lambda_{\min}, \lambda_{\max} \geq 0$ *independent of* t*.*

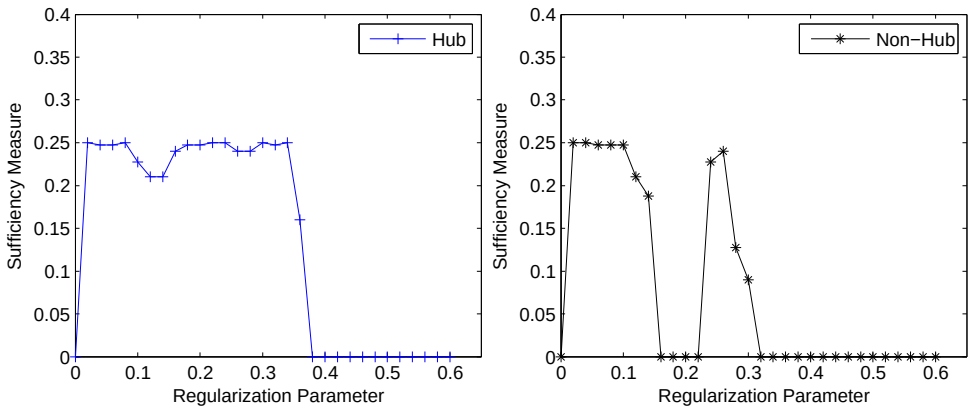
(c) For any $t \in V \setminus r$*,* $\exists\ t' \notin \mathcal{N}^*(r) : \lambda_{\min}(t') < \lambda_{\max}(t)$*.*

Additionally, $p_{r,b,\lambda}(t)$ *is a continuous function of* λ*.*

To build intuition for the assumptions, as well as our analysis in the sequel, it will be instructive to consider the behavior of the inclusion probability $p_{r,b,\lambda}(t)$ as we increase λ from zero to infinity. When λ is zero, the ℓ_1-estimator reduces to the unregularized conditional MLE: any variable $t \in V\backslash r$ will always occur in the neighborhood estimate of node r, and $p_{r,b,\lambda}(t)$ will be equal to one. As λ increases, the inclusion probability in turn reduces, and at a very large value of λ, the inclusion probability $p_{r,b,\lambda}(t)$ will become equal to zero:

this follows from the property of the ℓ_1-estimator, where there exists a large regularization weight when the parameter estimate becomes equal to zero.

In the assumptions above, it can be seen that if $\lambda_{\min}(t)$ and $\lambda_{\max}(t)$ exist, then by definition, we must have $\lambda_{\min}(t) \leq \lambda_{\max}(t)$. Part (a) of the assumption is a smoothness constraint that ensures that if the probability of inclusion or exclusion of a variable into a neighbourhood gets close to 1, then it stays close to 1, and does not vary wildly. Part (b) ensures that ranges of $[\lambda_{\min}(t), \lambda_{\max}(t)]$ intersect at least for all irrelevant variables $t \notin \mathcal{N}^*(r)$. This is a very mild assumption that ensures that the inclusion probability of an irrelevant variable does not stay exactly at one as we increase λ, and reduces at least very slightly (below the threshold of $1 - 2\exp(-c \log p)$) before other irrelevant variables have their inclusion probability drop from one all the way to zero. Part (c) is a closely related mild assumption that ensures that the probability of inclusion of atleast one irrelevant variable would have dropped by a small value from 1 before any other variable has its inclusion probability drop from one all the way to zero. We note that these mild technical assumptions on the inclusion probabilities always hold in our empirical observations.



(a) $\mathcal{M}_{r,b,\lambda}$ for a hub node ($degree = 19$)

(b) $\mathcal{M}_{r,b,\lambda}$ for a non-hub node ($degree = 1$)

Figure 2.1: Behaviour of $\mathcal{M}_{r,b,\lambda}$ for non-hub nodes and hub-nodes in a star graph on $p = 100$ nodes.

Armed with these assumptions, we now analyze the behavior of our sufficiency measure $\mathcal{M}_{r,b,\lambda}$. Our next proposition shows that there exists atleast one "bump" in the graph of the sufficiency measure against the regularization penalty λ.

Proposition 2.2 ("First Bump"). *Suppose Assumption 2.1 holds with constant $c > 0$. Let*

$$\gamma = 2\exp(-c\log p)\left(1 - 2\exp(-c\log p)\right). \tag{2.19}$$

For any node $r \in V$, let $\lambda_l = \inf\{\lambda \geq 0 : \mathcal{M}_{r,b,\lambda} \geq \gamma\}$ be the smallest regularization penalty where the sufficiency measure is greater than a small threshold above zero, and $\lambda_u = \inf\{\lambda > \lambda_l : \mathcal{M}_{r,b,\lambda} < \gamma\}$ be the next value of the regularization penalty where the sufficiency measure falls below that threshold. Then,

(a) the infima above are attained at finite values, and (b) for any $k \in (\gamma, 1/4]$, $\exists \lambda \in (\lambda_l, \lambda_u)$ *s.t.* $\mathcal{M}_{r,b,\lambda} \geq k$.

Our next two propositions track the behavior of the ℓ_1-estimate $\widehat{\mathcal{N}}_{b,\lambda}(r; D)$ after the first bump outlined above. The very next proposition provides the behavior for "non-hub" nodes.

Proposition 2.3 (Behavior at λ_u for "non-hub nodes")**.** *Let* $r \in V$ *be a "non-hub node" w.r.t. b samples, and constants for all conditions of Theorem 2.1 being* c_1, c_2, c_3, c_4. *Let Assumption 2.1 hold for a constant* $c > 1$ *with* $c < c_1 c_4$, *and let* λ_u *be as defined in Proposition 2.2. Then,* $\widehat{\mathcal{N}}_{b,\lambda_u}(r; D)$ *recovers the neighborhood with high probability:*

$$\begin{aligned} &\mathbb{P}\left(\mathcal{N}^*_{sub}(r) \subseteq \widehat{\mathcal{N}}_{b,\lambda_u}(r; D) \subseteq \mathcal{N}^*(r)\right) \\ &\qquad > 1 - 2\exp\left(-(c-1)\log p\right), \end{aligned}$$

where $\mathcal{N}^*_{sub}(r) = \left\{ t \in \mathcal{N}^*(r) \,\middle|\, |\theta^*_{rt}| \geq c_3\sqrt{d}\lambda \right\}$.

The proposition thus tells us that for "non-hub nodes", after the first bump when the value of $\mathcal{M}_{r,b,\lambda}$ becomes very close to zero, the ℓ_1-estimator recovers $\mathcal{N}^*_{\text{sub}}(r)$ w.h.p. (as also indicated by Theorem 2.1). Note that when increasing λ further, there would be further bump(s): the value of $\mathcal{M}_{r,b,\lambda}$ would rise, but would again drop back to zero: when λ is very large, the neighborhood estimate is null, so that the probability for any node to be in the neighborhood will be exactly zero; so that the sufficiency measure will be equal to zero. Figure 2.1b demonstrates this behavior in a simulated dataset.

On the other hand, for "hub nodes", the behavior of $\widehat{\mathcal{N}}_{b,\lambda}(r; D)$ at $\lambda = \lambda_u$, defined in Proposition 2.2, is given by the following proposition.

Proposition 2.4 (Behavior at λ_u for "hub nodes"). *Let $r \in V$ be a "hub node" w.r.t. b samples. Also, let Assumption 2.1 hold with constant $c > 1$. Then $\widehat{\mathcal{N}}_{b,\lambda_u}(r; D)$ excludes irrelevant variables with high-probability:*

$$\mathbb{P}\left(\widehat{\mathcal{N}}_{b,\lambda_u}(r; D) \subseteq \mathcal{N}^*(r)\right) > 1 - 2\exp\left(-(c-1)\log p\right).$$

The proposition thus tells us that for "hub nodes", after the first bump when the value of $\mathcal{M}_{r,b,\lambda}$ becomes very close to zero, irrelevant variables are excluded, though however there is no guarantee on relevant variables being included. Empirically in fact, the end of the first bump typically occurs at a very large value of λ when *all* variables are excluded; in particular, the graph of $\mathcal{M}_{r,b,\lambda}$ against λ typically has a single bump. Figure 2.1a demonstrates this behavior in a simulated dataset.

Propositions 2.3 and 2.4 thus motivate using the behaviors of the sufficiency measure as outlined above to distinguish hub nodes and non-hub nodes; and then compute the graph estimate using the neighborhood estimates from the non-hubs alone. This natural procedure is described in Algorithm 2.2.

The following theorem is a natural corollary of Theorem 2.1, and Propositions 2.3 and 2.4. Note that in the below, we assume that the true parameter set θ^* satisfies the incoherence conditions, $(A1)$ and $(A2)$, for all nodes $r \in V$;

Algorithm 2.2 Algorithm to compute neighborhood estimate $\widehat{\mathcal{N}}(r)$, for each node $r \in V$, and the overall edge estimate $\widehat{E}$

Input : Data $D := \{x^{(1)}, \ldots, x^{(n)}\}$, Regularization parameters $\Lambda := \{\lambda_1, \ldots, \lambda_s\}$, Sub-sample size b, No. of sub-samples N, Thresholds on sufficiency measure t_l and t_u, Node $r \in V$

Output: An estimate $\widehat{\mathcal{N}}(r)$ of the neighborhood for each $r \in V$, and the overall edge estimate $\widehat{E}$

foreach $r \in V$ **do**

$\forall\, \lambda \in \Lambda$, Compute $\widehat{\mathcal{M}}_{r,b,\lambda}(D)$ using Algorithm 2.1

$\lambda' \leftarrow$ Smallest $\lambda \in \Lambda$ s.t. $\widehat{\mathcal{M}}_{r,b,\lambda}(D) > t_u$

$\Lambda \leftarrow \{\lambda \in \Lambda : \lambda > \lambda'\}$

$\lambda_0 \leftarrow$ Smallest $\lambda \in \Lambda$ s.t. $\widehat{\mathcal{M}}_{r,b,\lambda}(D) < t_l$

$\widehat{\mathcal{N}}(r) \leftarrow \left\{ t \,|\, \hat{p}_{r,b,\lambda_0}(t; D) \geq \frac{1+\sqrt{1-4t_l}}{2} \right\}$

$\widehat{E} \leftarrow \bigcup_{r \in V} \{(r,t) \,|\, t \in \widehat{\mathcal{N}}(r)\}$

and that Assumption 2.1 holds $\forall\, r \in V$, with an appropriate constant $c > 2$, satisfying conditions of Proposition 2.3 for "non-hub nodes".

Theorem 2.2 (Guarantee for the estimator of Algorithm 2.2)**.** *Suppose we run Algorithm 2.2 setting* $t_l = 2\exp(-c\log p)(1-2\exp(-c\log p)) + \epsilon$, $t_u = 1/4 - \epsilon$, *the sub-sample size* $b = f(n)$ *(with* $\sqrt{n} \leq f(n) < n/2$*), and number of sub-samples* $N \geq \lceil n/f(n) \rceil$, *such that*

$$n > 18 f(n) \left[\log p + \log\left(4/\delta\right)\right] / \epsilon^2. \tag{2.20}$$

For any degree-value $d \in \{1, \ldots, p\}$ *and constant* $c'' > 0$, *denote*

$$E_d = \left\{ (s,t) \in E^* \,\middle|\, \min(d(s), d(t)) \leq d,\ |\theta^*_{st}| \geq c'' \sqrt{\frac{d \log p}{n}} \right\} \tag{2.21}$$

where $d(v)$ *corresponds to the degree of vertex* v *in* E^*. *Then, there exist*

constants c, c', c'', c'''*, such that if the sub-sample size scales as*

$$f(n) > c' d^3 \log p, \tag{2.22}$$

then the graph structure estimate $\widehat{E}$ *of Algorithm 2.2 satisfies:*

$$\mathbb{P}\left(E_d \subseteq \widehat{E} \subseteq E^*\right) \geq 1 - 2\exp\left(-c''' \log p\right) - \delta. \tag{2.23}$$

Now, let us define the *critical degree*, d_c, of a graph $G^* = (V, E^*)$, as the minimum degree such that neighborhoods of vertices with at most the said degree cover the whole graph, *i.e.*

$$\begin{aligned} d_c = &\min \quad d \\ &\text{s.t. } \forall\,(s,t) \in E^*, \text{ either } d(s) \leq d \text{ or } d(t) \leq d. \end{aligned} \tag{2.24}$$

The following corollary then gives the sample complexity for exact recovery of the graph, assuming that the edges have sufficient weight.

Corollary 2.2. *Let the conditions of Theorem 2.2 be satisfied, with* $b = f(n)$ *as the sub-sample size. Let* d_c *be the critical degree of the graph* G^**. Then there exist constants* c', c'', c''' *s.t. if the sub-sample size scales as*

$$f(n) > c' d_c^3 \log p, \tag{2.25}$$

and $|\theta^*_{st}| \geq c'' \sqrt{\frac{d_c \log p}{n}}\, \forall\,(s,t,) \in E^*$*, then*

$$\mathbb{P}\left(\widehat{E} = E^*\right) \geq 1 - 2\exp(-c''' \log p), \tag{2.26}$$

where $\widehat{E}$ *is the graph structure estimate from Algorithm 2.2.*

Note that we may choose $f(n) = c'n^{1-\rho}$, for some value of $\rho \in (0, 0.5]$, as the sub-sample size. The choice of ρ would be governed by d_c for the graph under consideration. For example, if d_c is a constant (e.g. $d_c = 1$ in a star graph), then the optimal choice of ρ would be 0.5, yielding a overall sample complexity of $\Omega\left((\log p)^2\right)$.

2.4 Experiments

In this section, we present experimental results demonstrating that our algorithm does indeed succeed in recovering graphs with a few hubs.

2.4.1 Synthetic Data

We first performed structure learning experiments on simulated data using 3 types of graphs:

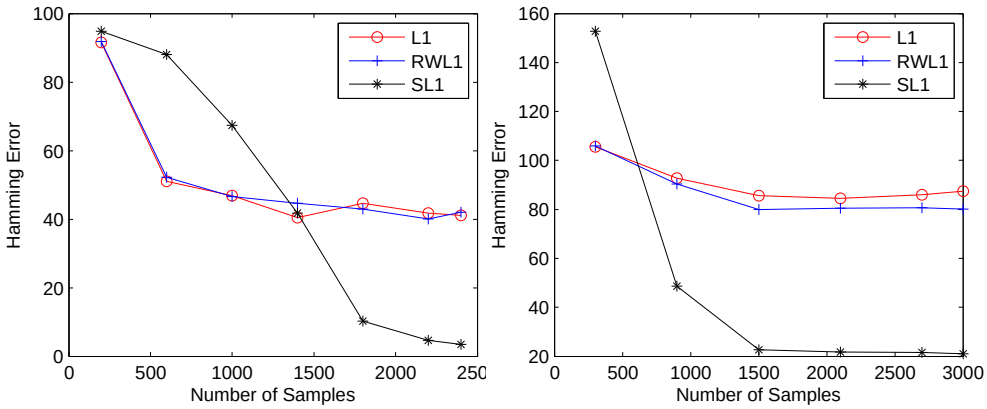
(a) a collection of stars with $p = 100$ nodes involving 5 hub nodes with degree $d = 19$, each connected to 19 other degree $d = 1$ nodes.

(b) a grid graph with 81 nodes (9×9), with 2 additional high degree hub-nodes of degree $d = 12$ (so that $p = 83$) attached to random points in the grid.

(c) a power-law graph on $p = 100$ nodes generated using the preferential attachment scheme [6].

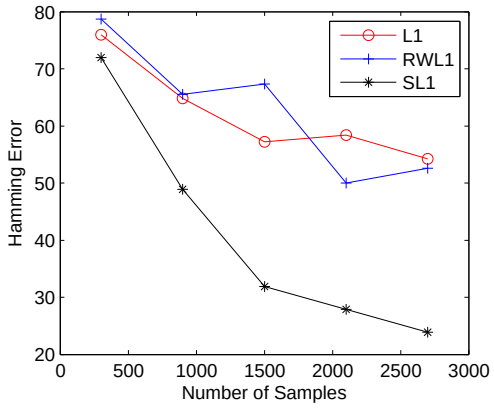
For each graph, we considered a pairwise Ising model with edge weight $\theta^*_{rt} = \frac{\omega}{\max(d_r, d_t)}$, for some $\omega > 0$, and where d_r and d_t were the degrees of r and t

respectively. For each such Ising model, we generated n i.i.d. samples $D = \{x^{(1)}, \ldots, x^{(n)}\}$ using Gibbs sampling.



(a) Stars Graph

(b) Hub+Grid Graph



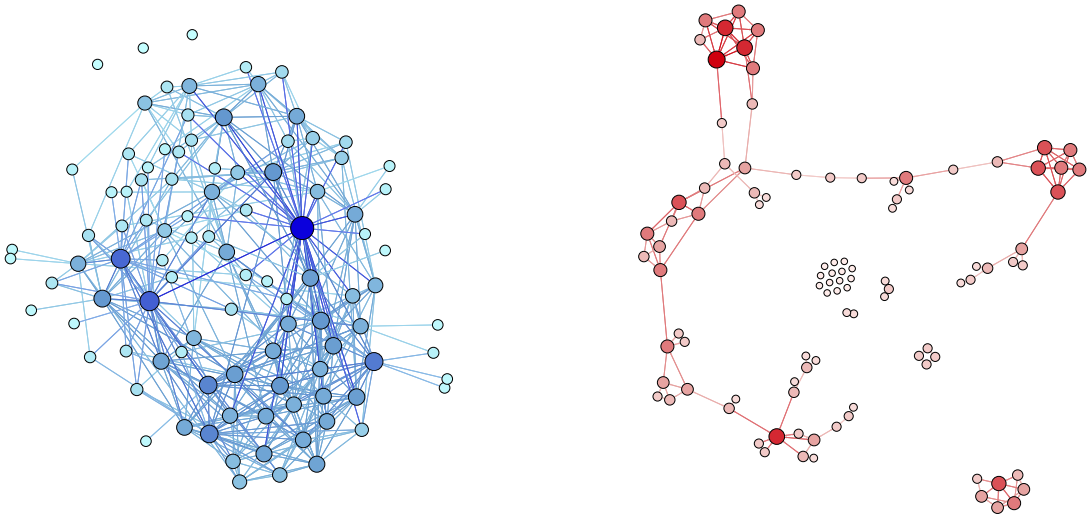
(c) Preferential Attachment Graph

Figure 2.2: Plots of Average Hamming Error vs Number of Samples

In all our experiments, for our algorithm (denoted as SL1 in our plots), the value of N, the number of times to subsample, was fixed to 60. We set lower and upper thresholds on the sufficiency measure as $t_l = 0.1$ and $t_u = 0.2$. The number of subsamples was set to $b = \min\left(20\sqrt{n}, \frac{n}{2}\right)$ and the set

of regularization parameters was taken as $\Lambda = \{0.005, 0.01, 0.015, \ldots, 1\}$. We performed comparisons with the ℓ_1-estimator [49] (denoted as L1 in our plots) and the reweighted ℓ_1-estimator for scale-free graphs [38] (denoted as RWL1 in our plots). For both these methods, the best regularization parameter was chosen using the Bayesian information criterion (BIC) from the grid of regularization parameters Λ. Figure 2.2 shows plots of the Average Hamming Error (*i.e.* average number of mismatches from the true graph) with varying number of samples for our method and the baselines, computed over an average of 10 trials. Since our estimate uses subsamples to compute its sufficiency measure, when the number of samples is extremely low, the deviation of the sample sufficiency measure estimate from the population sufficiency measure becomes large enough so that the resulting mistakes made by our method in designating hubs and non-hubs increase its overall Hamming error. We note however that at such extremely low number of samples, it can be seen that the overall Hamming error of any estimator is quite high, so that none of the estimators provide useful graph estimates in any case. It can be seen that other than at such extremely few samples, we achieve *much lower* Hamming error than both L1 and RWL1, and which is particularly pronounced for scale-free graphs such as those provided by the preferential attachment model.

2.4.2 Real Data



(a) L1 with BIC

(b) Our Algorithm

Figure 2.3: Graphs obtained using US Senate voting records data from the 109th congress [5]

We ran our algorithm on a data set consisting of US senate voting records data from the 109th congress (2004 - 2006) [5]. It consists of 100 nodes ($p = 100$), corresponding to 100 senators. There are 542 samples, each representing a bill that was put to vote. For each *(senator, bill)* pair, the vote is recorded as either a 1 (representing a *yes*), a -1 (representing a *no*) or a 0 (representing a *missed vote*). For the purpose of the experiment, all 0 entries were replaced by -1, as also done in [5].

Our algorithm was run with the parameters $N = 60, t_l = 0.1, t_u = 0.2, b = 450$ and $\Lambda = \{0.005, 0.01, 0.015, \ldots, 1\}$. Figure 2.3b shows the graph obtained using our method, whiles Figure 2.3a shows the graph obtained by running the ℓ_1-estimator [49] with the regularization parameter being chosen using the Bayesian Information Criterion (BIC) from the set of regularization parameters Λ.

We see that the graph obtained using the ℓ_1-estimator with BIC is much denser than what we obtain. This also corroborates the observation of [37], that BIC leads to larger density in high dimensions. A few of the nodes in the graph using our algorithm are seen to have 0 degree, and are thus disconnected from the graph. This might be because these might be higher degree "hub" nodes, but for which the number of samples is not sufficient enough to provide a reliable estimate of the neighbourhoods vis-à-vis their degree. Overall, the sparse graph we obtained using our reliability indicator based method suggests the need for such reliability indicators to prevent the inclusion of spurious edge-associations.

Chapter 3

Kernel Ridge Regression via Partitioning[2]

3.1 Introduction

Kernel methods find wide and varied applicability in machine learning. One such application of kernels is the problem of Kernel Ridge Regression (KRR). Given covariate-response pairs (x, y), the goal is to compute a kernel-based function f such that $f(x)$ approximates y well on average. In this regard, several learning methods with different kernel classes have been shown to achieve good predictive performance. Despite their good generalization, kernel methods suffer from a computational drawback if the number of samples n is large — which is more so the case in modern settings. They require at least a computational cost of $O(n^2)$, which is the time required to compute the kernel matrix, and $O(n^3)$ time when the kernel matrix also has to be inverted, which is the case for KRR.

Several approaches have been proposed to mitigate this, including Nyström approximations [2, 4, 51], approximations via random features [15, 45, 46, 63],

[2]This chapter is based on [59]. The author of this work was the first author and primary contributor to [59]. The author proposed the algorithm considered in this chapter, and analyzed it in a general setting as well as for specific kernels. The author also collaborated on writing the paper, as well as performing the empirical tests.

and others [48, 62]. While these approaches help computationally, they typically incur an error over-and-above the error incurred by a KRR estimate on the entire data. Another class of approaches that may not incur such an error are based on what we loosely characterize as divide-and-conquer approaches, wherein the data points are *divided* into smaller sets, and estimators trained on the divisions. These approaches may further be categorized into three main classes: division by uniform splitting [67], division by clustering [24, 28] or division by partitioning [23]. The latter may also include local learning approaches, which are based on estimates using training points near a test point [8, 25, 52, 65]. Given this considerable line of work, there is now an understanding that these divide-and-conquer approaches provide computational benefits, and yet have statistical performance that is either asymptotically equivalent, or at most slightly worse than that of the whole KRR estimator. Please see [23, 28, 67] and references therein for results reflecting this understanding for uniform splitting, clustering and partitioning respectively. However, these results have restrictive assumptions, applicability or other limitations, such as requiring the covariates/responses to be bounded [23], or only being applicable to specific kernels e.g. Gaussian [23] or linear [24], or only being targeted to classification [24, 28], or providing error rates only on the training error [28]. Moreover, approaches based on uniform splitting, such as [67], can suffer from worse *approximation* error, as alluded to shortly.

In this work, we consider a partitioning based divide-and-conquer approach to kernel ridge regression. We provide a refined analysis, applicable

to general kernels, which leads us to this surprising conclusion: the partitioning based approach not only has computational benefits outlined in previous papers, but also has strong statistical benefits when compared to the whole KRR estimator. In other words, based on both a statistical and computational viewpoint, we are able to recommend the use of the partitioning based approach over the whole KRR approach.

The partitioning based approach is: Given n sample points, we divide them into m groups based on a fixed disjoint partitioning of input space $\mathcal{X}$ that the samples are drawn from. One way to obtain this partition is via clustering, however, in principle, any partition that satisfies certain assumptions (detailed in Section 3.4.1) would be acceptable. Once the samples have been divided, we learn a kernel ridge regression estimate for each partition using only its own samples. The conquering step *i.e.* computing the overall estimator, $\hat{f}_C$, is then simple: Each individual estimator is applied to its respective partition. Thus, to perform prediction for a new point, we simply identify its partition, and use the estimator for that partition. Now, partitioning has a clear computational advantage since each estimate is trained over only a fraction of the points. Moreover, partitioning may provide statistical advantages as well if there is an inherent approximation error in the problem *i.e.*, the true regressor function, f^*, lies outside the space of kernel-based functions. In this case, the KRR estimator on the whole data, say $\hat{f}_{whole}$, or the KRR estimator based on uniform splitting, say $\hat{f}_{avg}$, both may be viewed as estimating the best single kernel-based function that approximates f^*. However, if we partition, then we

are estimating the best m-piece-wise kernel-based function to approximate f^*. Indeed, we can show that the approximation error for $\hat{f}_C$ is lesser than $\hat{f}_{avg}$, and corroborate this experimentally. The residual error terms on the other hand are typically of the same order, so that the overall generalization error for our method is lower. In addition, there is yet another potential computational advantage of partitioning: prediction is faster since for a new point, the kernel values must be computed w.r.t. only a fraction of the points (as opposed to all the points for $\hat{f}_{whole}$ or $\hat{f}_{avg}$).

3.1.1 Related Work

We briefly review some of the earlier mentioned work that provide theoretical analyses of divide and conquer approaches, based on uniform splitting, and partitioning. [67] analyze the uniform splitting approach where the samples are split uniformly at random, followed by an averaging of the KRR estimate of each split. The authors have derived generalization rates for this estimator, and matched optimal rates as long as the number of splits is not *large*, and the true function f^* lies in the specified space of kernel-based functions. However, as mentioned previously, such an estimator can have worse approximation error than our estimator, $\hat{f}_C$, when the true function, f^*, lies outside the space of kernel-based functions. [23] analyze a partition based approach as in this work: their estimator works by partitioning the input space, and predicting using KRR/SVM estimates over each partition individually. For this estimator, [23] derive generalization rates when using Gaussian

kernels, and under additional restrictions: they require bounded covariates, $\|x\| \leq B$, bounded response, $|y| \leq M$, and that each partition be bounded by a ball of suitable radius, R.[1] Given these restrictions, they show suitable choices for R and the Gaussian kernel scale γ which yield optimal rates when the true function f^* lies in a smooth Sobolev/Besov space. In contrast, we provide a more general analysis that does not enforce a bound on the covariates, response, or the size of the partition. Moreover, we are able to apply it to kernels other than the Gaussian kernel, and achieve minimax optimal rates when the true function, f^*, lies in the space of kernel-based functions. When it doesn't, we provide an oracle inequality similar to [23], which could then be specialized to obtain similar rates for their specific setting. More importantly, our analysis is also able to show that in general, the approximation component of this inequality is lesser than the approximation component of the whole KRR estimator, while the residual components can be of the same order.

From a theoretical standpoint, the generalization error for KRR has been studied extensively — an incomplete list includes [10, 14, 22, 29, 54, 56, 66]. We shall not delve into a comparison among these, but instead refer the interested reader to [29, Section 2.5], [22, Section 3], for more details. Of relevance to our analysis is the approach in [29], wherein the generalization error is broken down into contributions of *regularization*, *bias* and *variance*,

[1] One way of obtaining such partitions, as suggested by the authors, is through the Voronoi partitioning of the input space.

and each of these is bounded separately. We adopt a similar strategy to control the expected error of our estimator, $\hat{f}_C$.

The rest of this chapter is organized as follows. Section 3.2 sets up notation and introduces the problem. Section 3.3 details our *DC*-estimator $\hat{f}_C$. Section 3.4 presents bounds on its generalization error. Section 3.5 instantiates these bounds for two kernel classes, and discusses the *approximation* component of $\hat{f}_C$. Finally, Section 3.6 provides empirical performance results. All proofs are can be found in Appendix C.

3.2 Notation and Preliminaries

Reproducing Kernel Hilbert Spaces. Consider any set $\mathcal{X}$, typically the space of the input data. A function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, is called a *kernel function* if it is *continuous, symmetric*, and *positive definite*. With any kernel function K, one can associate a unique Hilbert space called the *Reproducing Kernel Hilbert Space* of K (abbreviated as RKHS henceforth). For $x \in \mathcal{X}$, let $\phi_x : \mathcal{X} \to \mathbb{R}$ be the function $\phi_x(\cdot) := K(x, \cdot)$. Then, the unique RKHS corresponding to kernel K, denoted as $\mathcal{H}$, is a Hilbert space of functions from $\mathcal{X}$ to $\mathbb{R}$ defined as: $\mathcal{H} := \overline{\text{span}}\{\phi_x\}$

Thus, any $f \in \mathcal{H}$ has the representation $f = \sum_j \alpha_j \phi_{x_j} = \sum_j \alpha_j K(x_j, \cdot)$ with $\alpha_j \in \mathbb{R}$, $\forall\, j$. The inner product on $\mathcal{H}$ is given as: $\langle \sum_j \alpha_j \phi_{x_j}, \sum_k \beta_k \phi_{x_k} \rangle_{\mathcal{H}} = \sum_j \sum_k \alpha_j \beta_k K(x_j, x_k)$. The inner product also induces a norm on $\mathcal{H}$, given as: $\|f\|_{\mathcal{H}} = \sqrt{\langle f, f \rangle_{\mathcal{H}}}$, for any $f \in \mathcal{H}$.

Kernel Ridge Regression. We are given a training set of n i.i.d. samples, $\mathbf{D} = \{x_i, y_i\}_{i=1}^n$. x (and x_i), also called the covariate, is a random variable in the input space $\mathcal{X}$ with distribution $\mathcal{P}$. y (and y_i) is a random variable in the output space $\mathcal{Y}$, also called the response. We consider $\mathcal{Y} \subseteq \mathbb{R}$ and assume an additive noise model for relating the response to the covariate:

$$y = f^*(x) + \eta, \tag{3.1}$$

where η is the random noise variable, and $f^* : \mathcal{X} \to \mathbb{R}$ is an unknown function. The goal of regression is to compute the function (or an approximation to) f^*. We also assume that the noise has zero mean and bounded variance, $\mathbb{E}[\eta|x] = 0$ and $\mathbb{E}[\eta^2|x] \leq \sigma^2$, and that f^* is square integreable with respect to the measure on $\mathcal{X}$ *i.e.* $f^* \in \mathcal{L}_2(\mathcal{X}, \mathcal{P}) := \{f : \mathcal{X} \to \mathbb{R} \mid \|f\|_{L_2}^2 = \mathbb{E}_{\mathcal{P}}[f(x)^2] < \infty\}$

A Kernel Ridge Regression (abridged as KRR) estimator approximates f^* by a function in the RKHS space $\mathcal{H}$ (corresponding to kernel K). We require that the RKHS space $\mathcal{H} \subset L_2(\mathcal{X}, \mathcal{P})$[2]. The KRR estimate $\hat{f}_\lambda \in \mathcal{H}$ is obtained by solving the optimization problem:

$$\hat{f}_\lambda = \arg\min_{f \in \mathcal{H}} \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \|f\|_{\mathcal{H}}^2 \tag{3.2}$$

where $\lambda > 0$ is the regularization penalty. This is tractable since, by the representer theorem, we have the relation $\hat{f}_\lambda = \sum_{i=1}^n \alpha_i \phi_{x_i}$, with $\alpha \in \mathbb{R}^n$ having the closed form expression: $\alpha = (G + n\lambda I)^{-1} y$, where $G \in \mathbb{R}^{n \times n}$ is the kernel matrix, *i.e.* $G_{ij} = K(x_i, x_j)$ $(i, j \in [n])$.

[2]This means $\forall x$, $\mathbb{E}_{y \sim \mathcal{P}}[K(x, y)^2] < \infty$ — which is always true for several kernel classes, including Gaussian, Laplacian, or any trace class kernel w.r.t. $\mathcal{P}$

Generalization/Prediction Error. For any estimator $\hat{f} : \mathcal{X} \to \mathbb{R}$, the generalization error quantifies closeness to f^*, by measuring the average prediction error. It is defined as:

$$\text{Err}(\hat{f}) := \mathbb{E}\left[(\hat{f}(x) - f^*(x))^2\right] = \left\|\hat{f} - f^*\right\|_{L_2}^2 \tag{3.3}$$

When the estimator is random, for example the KRR estimate $\hat{f}_\lambda$ in Eq. (3.2) depends on random samples, we may quantify the average error over the randomness *i.e.* bound $\mathbb{E}_D[\text{Err}(\hat{f}_\lambda)]$, where the expectation is taken over the samples $\mathbf{D}$. We provide bounds on the quantity $\mathbb{E}_D[\text{Err}(\hat{f}_C)]$, where $\hat{f}_C$ is our *divide-and-conquer* estimator (DC-estimator) described in Section 3.3.

Partition-specific notation. Since our estimator, $\hat{f}_C$, is based on partitioning, we setup some notation here for partition-specific quantities that play a role throughout the analysis. We say that the input space $\mathcal{X}$ has a disjoint partition $\{C_1, \ldots, C_m\}$ if: $\mathcal{X} = \cup_{i=1}^m C_i$, and $C_i \cap C_j = \{\phi\}\, \forall\, i, j \in [m], i \neq j$. Given data $\mathbf{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, we define a partition-based empirical covariance operator as: $\hat{\Sigma}_i = \frac{1}{n}\sum_{j=1}^n (\phi_{x_j} \otimes \phi_{x_j}) \mathbb{1}\left(x_j \in C_i\right)$, where $\mathbb{1}\left(\cdot\right)$ denotes the indicator function and $\phi_x \otimes \phi_x$ denotes the operator $\phi_x \left\langle \phi_x, \cdot \right\rangle_{\mathcal{H}}$. We define its population counterpart as: $\Sigma_i = \mathbb{E}\left[(\phi_x \otimes \phi_x) \mathbb{1}\left(x \in C_i\right)\right]$. Note the relation: $\Sigma = \sum_{i=1}^m \Sigma_i$, where $\Sigma = \mathbb{E}\left[\phi_x \otimes \phi_x\right]$ is the overall covariance operator.

We let $\{\lambda_j^i, v_j^i\}_{j=1}^\infty$ denote the collection of eigenvalue-eigenfunction pairs for Σ_i. For any $\lambda > 0$, we define a spectral sum for Σ_i, $S_i(\lambda) = \sum_j \frac{\lambda_j^i}{\lambda_j^i + \lambda}$. Similarly, letting $\{\lambda_j, v_j\}_{j=1}^\infty$ be the eigenvalue-eigenfunction pairs for the overall

covariance Σ, the corresponding sum for Σ is defined as $S(\lambda) = \sum_j \frac{\lambda_j}{\lambda_j + \lambda}$. The quantity $S(\lambda)$ has appeared in previous work on KRR [29, 66, 67], and is called the *effective dimensionality* of the kernel K (at scale λ). Typically, it plays the same role as dimension does in finite dimensional ridge regression. We shall refer to the quantity $S_i(\lambda)$ as the *effective dimensionality* of partition C_i. Finally, we let $p_i = \mathbb{P}(x \in C_i)$ denote the probability mass of partition C_i.

3.3 The DC-estimator: $\hat{f}_C$

When the number of samples n is large, solving Eq. (3.2) may be computationally prohibitive, requiring $O(n^3)$ time in the worst case. A simple strategy to tackle this is by *dividing* the samples $\mathbf{D}$ into disjoint partitions, and computing an estimate separately for each partition. In this work, we consider partitions of $\mathbf{D}$ which adhere to an underlying disjoint partition of the input space $\mathcal{X}$. Suppose that the input space $\mathcal{X}$ has a disjoint partition $\{C_1, \ldots, C_m\}$. Note that m denotes the number of partitions. Also, suppose that given any point $x \in \mathcal{X}$, we can find the partition it belongs to from the set $\{C_1, \ldots, C_m\}$.

Now, we divide the data set $\mathbf{D}$ in agreement with this partitioning of $\mathcal{X}$ *i.e.* we split $\mathbf{D} = \{D_1, \ldots, D_m\}$ with $D_i = \{(x_j, y_j) \,|\, x_j \in C_i,\, j = 1, \ldots, n\}$. Let $|D_i| = n_i$. Then, for any partition $i \in [m]$, we compute a local estimator using only the points in its partition:

$$\hat{f}_{i,\lambda} = \operatorname*{arg\,min}_{f \in \mathcal{H}} \frac{1}{n_i} \sum_{j:\,(x_j, y_j) \in D_i} (y_j - f(x_j))^2 + \lambda \left\| f \right\|_{\mathcal{H}}^2 \tag{3.4}$$

where $\lambda > 0$ is the regularization penalty. Finally, the overall estimator, $\hat{f}_C$, comprises of the local estimators applied to their corresponding partitions:

$$\hat{f}_C(x) = \hat{f}_{i,\lambda}(x) \text{ if } x \in C_i \tag{3.5}$$

In practice, one can use a clustering algorithm to cluster the points in $\mathbf{D}$, as well as determine membership for new points x.

3.4 Generalization Error of $\hat{f}_C$

In this section we quantify the error $\mathbb{E}_D\left[\text{Err}(\hat{f}_C)\right]$, where $\hat{f}_C$ is the *DC*-estimator from Eq. 3.5. First, we observe that $\text{Err}(\hat{f}_C)$ can be decomposed as a sum of errors of the local estimators, $\hat{f}_{i,\lambda}$, on their corresponding partitions C_i, $i \in [m]$. We have:

$$\text{Err}(\hat{f}_C) = \mathbb{E}\left[(f^*(x) - \hat{f}_C(x))^2\right] = \sum_{i=1}^{m} \mathbb{E}\left[(f^*(x) - \hat{f}_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] = \sum_{i=1}^{m} \text{Err}_i(\hat{f}_{i,\lambda}) \tag{3.6}$$

where $\mathbb{1}\left(\cdot\right)$ denotes the indicator function, and we have defined the partition-wise error: $\text{Err}_i(\hat{f}_{i,\lambda}) := \mathbb{E}[(f^*(x) - \hat{f}_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)]$. By linearity of expectation, $\mathbb{E}_D\left[\text{Err}(\hat{f}_C)\right] = \sum_{i=1}^{m} \mathbb{E}_D\left[\text{Err}_i(\hat{f}_{i,\lambda})\right]$. Therefore, to obtain a bound on $\mathbb{E}_D\left[\text{Err}(\hat{f}_C)\right]$ we need to bound $\mathbb{E}_D\left[\text{Err}_i(\hat{f}_{i,\lambda})\right]$, for every $i \in [m]$. Now, to control $\mathbb{E}_D\left[\text{Err}_i(\hat{f}_{i,\lambda})\right]$, we bound it as a sum of intermediate error terms[3], and in turn bound these intermediate terms. For this purpose, we define the

[3]Similar to the usual bias-variance decomposition; or the decomposition in [29, 67]. In contrast, loosely speaking, [23] analyze the error of $\hat{f}_C$ by viewing it as a Standard KRR with a new kernel $K1(x, x') = \sum_{i=1}^{m} K(x, x')\mathbb{1}\left(x \in C_i\right)\mathbb{1}\left(x' \in C_i\right)$

following estimates (for each $i \in [m]$):

$$\text{For any } \overline{\lambda} \geq 0, \quad f_{i,\overline{\lambda}} = \operatorname*{arg\,min}_{f \in \mathcal{H}} \mathbb{E}\left[(y - f(x))^2 \,|\, x \in C_i\right] + \overline{\lambda}\,\|f\|_{\mathcal{H}}^2 \tag{3.7}$$

$$f_{i,\lambda} = \operatorname*{arg\,min}_{f \in \mathcal{H}} \mathbb{E}\left[(y - f(x))^2 \,|\, x \in C_i\right] + \lambda\,\|f\|_{\mathcal{H}}^2 \tag{3.8}$$

$$\bar{f}_{i,\lambda} = \mathbb{E}_D[\hat{f}_{i,\lambda}] \tag{3.9}$$

$f_{i,\overline{\lambda}}$ and $f_{i,\lambda}$ are the optimal *population* KRR estimates for partition C_i, with regularization penalties $\overline{\lambda}$ and λ respectively. $\bar{f}_{i,\lambda}$ is the expected value of the *empirical* KRR estimate from Eq. (3.4), with the expectation taken over the samples $\mathbf{D}$. Note that there is no source of randomness in all of the above quantities, whereas $\hat{f}_{i,\lambda}$ is a random quantity due to its dependence on the random samples $\mathbf{D}$. Now, based on the above estimates, we define the following error terms:

Definition 2. *For any $\lambda > 0$ and $\overline{\lambda} \in [0, \lambda]$, we define*

$$\textbf{Approximation Error} \;:\; \mathit{Approx}_i(\overline{\lambda}) = \mathbb{E}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \tag{3.10}$$

$$\textbf{Regularization Error} \;:\; \mathit{Reg}_i(\overline{\lambda}, \lambda) = \mathbb{E}\left[(f_{i,\overline{\lambda}}(x) - f_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \tag{3.11}$$

$$\textbf{Bias} \;:\; \mathit{Bias}_i(\lambda, n) = \mathbb{E}\left[(f_{i,\lambda}(x) - \bar{f}_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \tag{3.12}$$

$$\textbf{Variance} \;:\; \mathit{Var}_i(\lambda, D) = \mathbb{E}\left[(\bar{f}_{i,\lambda}(x) - \hat{f}_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \tag{3.13}$$

The intent of $f_{i,\overline{\lambda}}$ is to correspond to the *best* kernel function that approximates f^* in the partition C_i. $\overline{\lambda}$ may be viewed as a *small* regularization penalty that trades-off the approximation error, $\text{Approx}_i(\overline{\lambda})$, to $\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}$ (which

influences other terms in Definition 2). Ideally, if the unknown function f^* lies in the RKHS space $\mathcal{H}$, a choice of $\overline{\lambda} = 0$ would suffice. In this case, we would have $f_{i,\overline{\lambda}} = f_{i,0} = f^*$ — implying zero approximation error *i.e.* $\text{Approx}_i(\overline{\lambda}) = \text{Approx}_i(0) = 0$, and bounded $\|f_{i,\overline{\lambda}}\|_{\mathcal{H}} (= \|f^*\|_{\mathcal{H}})$. The following lemma describes the decomposition of $\mathbb{E}_D\left[\text{Err}_i(\hat{f}_{i,\lambda})\right]$.

Lemma 3.1 (Error Decomposition)**.** *For each partition $i \in [m]$, the error $\mathbb{E}_D\left[Err_i(\hat{f}_{i,\lambda})\right]$ decomposes as (for any $\overline{\lambda} \in [0, \lambda]$):*

$$\mathbb{E}_D\left[Err_i(\hat{f}_{i,\lambda})\right] \leq 2\left[Approx_i(\overline{\lambda}) + 2Reg_i(\overline{\lambda}, \lambda) + 2Bias_i(\lambda, n) + 2\mathbb{E}_D\left[Var_i(\lambda, D)\right]\right] \tag{3.14}$$

Thus, the overall error $\mathbb{E}_D\left[Err(\hat{f}_C)\right]$ decomposes as (for any $\overline{\lambda} \in [0, \lambda]$):

$$\mathbb{E}_D\left[Err(\hat{f}_C)\right] \leq 2\left[\sum_{i=1}^{m} Approx_i(\overline{\lambda}) + 2\sum_{i=1}^{m} Reg_i(\overline{\lambda}, \lambda) + 2\sum_{i=1}^{m} Bias_i(\lambda, n) + 2\sum_{i=1}^{m} \mathbb{E}_D\left[Var_i(\lambda, D)\right]\right] \tag{3.15}$$

In the above decomposition we have considered the same choice of λ (and $\overline{\lambda}$) for all partitions, a similar decomposition would hold even if we were to choose a different λ (and $\overline{\lambda}$) for each partition.

To summarize, in Lemma 3.1, we have decomposed the overall error of our estimator, $\mathbb{E}_D\left[\text{Err}(\hat{f}_C)\right]$, as a sum of errors over each partition, which have further been broken into four components: *Approximation, Regularization, Bias* and *Variance*. The rest of this section deals with bounding these terms. First, we require certain assumptions on the partitions. These are provided in Section 3.4.1. Then, Section 3.4.2 discusses the bounds on the component terms, for any partition.

3.4.1 Assumptions

In this section, we state three assumptions needed to bound the terms in Lemma 3.1. It may be useful at this point to recall partition-specific definitions from Section 3.2. We also remark that two of our assumptions are fairly standard (Assumption 3.1 and Assumption 3.2), and analogous versions have appeared before [23, 29, 67]. The last assumption, Assumption 3.3, is novel. However, we have validated it extensively on both real and synthetic data sets (see Section 3.6).

Now, our first assumption concerns the existence of higher-order moments of the eigenfunctions, v_j^i.

Assumption 3.1 (Eigenfunction moments)**.** *Let* $\{\lambda_j^i, v_j^i\}_{j=1}^{\infty}$ *denote the eigenvalue-eigenfunction pairs for the covariance operator* Σ_i*. Then,* $\forall i \in [m]$*,* $\forall j$ *s.t.* $\lambda_j^i \neq 0$*, and for some constant* $k \geq 2$*, we assume* $\mathbb{E}\left[\left(v_j^i(x)^2 \mathbb{1}\left(x \in C_i\right)/\lambda_j^i\right)^{2k}\right] \leq a_1^k$*, where* a_1 *is a constant.*

Assumption 3.1 requires *sufficiently* many higher moments of $\left(v_j^i(x)^2 \mathbb{1}\left(x \in C_i\right)/\lambda_j^i\right)$ to exist. This assumption may also be interpreted as requiring partition-wise sub-Gaussian behaviour (up to $2k$ moments) in the RKHS space, given its primary application to the bounds (see Appendix B.2 for more details). Note that this assumption is similar to [67, Assumption A], but applied to each partition.

Our next assumption concerns the approximation variable $(f^*(x) - f_{i,\overline{\lambda}}(x))$, requiring its fourth moment to be bounded.

Assumption 3.2 (Finite Approximation)**.** $\forall i \in [m]$*, and any* $\overline{\lambda} \geq 0$*, we assume there exists a constant* $A_i(\overline{\lambda}) \geq 0$ *such that* $\mathbb{E}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))^4 \,|\, x \in C_i\right] \leq A_i(\overline{\lambda})^4$*, where* $f_{i,\overline{\lambda}}$ *is the solution of the optimization problem in Eq. 3.7.*

While the above assumption is stated for any $\overline{\lambda}$, we really only care about the actual $\overline{\lambda}$ used in Eq. 3.14. For example, if $f^* \in \mathcal{H}$, then as noted earlier, a choice of $\overline{\lambda} = 0$ suffices — and in this case, Assumption 3.2 trivially holds with $A_i(\overline{\lambda}) = 0$ at $\overline{\lambda} = 0$, since $f_{i,\overline{\lambda}} = f^*$ at $\overline{\lambda} = 0$.

Our final assumption enforces that the sum of *effective dimensionality* over all the partitions be bounded in terms of the overall *effective dimensionality*. We define the **goodness measure** of a partition $\{C_1, \ldots, C_m\}$ as: $g(\lambda) := \frac{\sum_{i=1}^m S_i(\lambda p_i)}{S(\lambda)}$. Now, we have the following assumption.

Assumption 3.3 (Goodness of Partition)**.** *Let* $\lambda > 0$ *be the regularization penalty in Eq.* (3.4)*, for any* $i \in [m]$*. Then, we require:* $g(\lambda) = O(1)$*.*

In Section 3.5, we show that if we have $g(\lambda) = O(1)$ for a λ decaying *suitably* in terms of n, the *DC*-estimator can achieve optimal minimax rates. In other words, if the partitioning preserves the overall *effective dimensionality*, then there is no loss in the generalization error. We validate the above assumption (at *suitable* λ) by estimating $g(\lambda)$ on real and synthetic data sets (see Section 3.6). $g(\lambda)$ may be viewed as a surrogate for the *suitability* of a partition for the DC-estimator, and can help guide the choice of partition.

3.4.2 Bounds on Reg_i, Bias_i and Var_i

We can now provide bounds on the terms $\text{Reg}_i(\lambda, \overline{\lambda})$, $\text{Bias}_i(\lambda, n)$, $\mathbb{E}_D\left[\text{Var}_i(\lambda, D)\right]$, occurring in Lemma 3.1, for any partition $i \in [m]$. In the interest of space, we refer the reader to Lemma B.1 in Appendix B, as well as the discussion there. In the next section, we use the bounds derived in these Lemmas to obtain overall generalization rates for specific kernels.

3.5 Bounds under Specific Cases

In this section, using the bounds on *regularization, bias* and *variance* (stated in Lemma B.1), we instantiate the overall error bounds for two different kernel classes. We do this under the assumption that $f^* \in \mathcal{H}$. When $f^* \notin \mathcal{H}$, we provide an oracle inequality for the error term and contrast this with a similar inequality derived in [67]. Throughout this section, we assume that the conditions of Lemma B.1.

3.5.1 $f^* \in \mathcal{H}$ — Zero approximation error

As mentioned earlier, in this case a choice of $\overline{\lambda} = 0$ suffices. With $\overline{\lambda} = 0$, we have $f_{i,\overline{\lambda}} = f^*$ (from Eq. (3.7)). Thus, $\text{Approx}_i(\overline{\lambda}) = 0$ at $\overline{\lambda} = 0$. Also, Assumption 3.2 trivially holds with $A_i(\overline{\lambda}) = 0$ at $\overline{\lambda} = 0$. Now, we provide overall generalization bounds for two kernel classes. We consider kernels with a finite rank — examples include the linear and polynomial kernels, and we consider kernels with exponentially decaying eigenvalues — an example here is the Gaussian kernel. An additional result for kernels with polynomial decaying

eigenvalues — of which sobolev kernels are an example — can be found in Appendix B.3.

Theorem 3.1 (Finite Rank Kernels)**.** *Let $f^* \in \mathcal{H}$ and suppose kernel K has a finite rank r. Let m denote the number of partitions. Then, the overall error for $\hat{f}_C$ is:*

$$\mathbb{E}_D\left[Err(\hat{f}_C)\right] = O\left(\lambda \left\|f^*\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{n} g(\lambda) S(\lambda) + m\left(\frac{r^2 \log r}{n}\right)^{k/2}\left(\left\|f^*\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{\lambda}\right)\right) \tag{3.16}$$

Now, if $m = O\left(\sqrt{\frac{n^{(k-4)}}{(r^2 \log r)^k}}\right)$ and Assumption 3.3 holds at $\lambda = r/n$, then $\hat{f}_C$ achieves the optimal rate: $\mathbb{E}_D\left[Err(\hat{f}_C)\right] = O\left(r/n\right)$ at $\lambda = r/n$.

Theorem 3.2 (Kernels with exponential eigenvalue decay)**.** *Let $f^* \in \mathcal{H}$ and suppose kernel K has eigenvalues that decay as: $\lambda_j \leq c_1 \exp(-c_2 j^2)$ ($\forall j$, and constants $c_1, c_2 > 0$). Let m denote the number of partitions. Then, the overall error for $\hat{f}_C$ is:*

$$\mathbb{E}_D\left[Err(\hat{f}_C)\right] = O\left(\lambda \left\|f^*\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{n} g(\lambda) S(\lambda) + m\left(\frac{\log n (\log\log n)}{n}\right)^{k/2}\left(\left\|f^*\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{\lambda}\right)\right) \tag{3.17}$$

Now, if $m = O\left(\sqrt{\frac{n^{(k-4)}}{(\log n \log\log n)^k}}\right)$ and Assumption 3.3 holds at $\lambda = 1/n$, then $\hat{f}_C$ achieves the optimal rate: $\mathbb{E}_D\left[Err(\hat{f}_C)\right] = O\left(\sqrt{\log n}/n\right)$ at $\lambda = 1/n$.

Note that the requirement of an upper-limit on m in the above theorems is only meaningful for a sufficiently large k, in particular $k \geq 4$. In other words, we would need at least 4 moments of the quantity in Assumption 3.1 to exist. If this is true, and if Assumption 3.3 holds, then Theorem 3.1 and Theorem 3.2 guarantee the rates $\mathbb{E}_D\left[\mathrm{Err}(\hat{f}_C)\right] = O\left(r/n\right)$ and $\mathbb{E}_D\left[\mathrm{Err}(\hat{f}_C)\right] = O(\sqrt{\log n}/n)$ — both of which are minimax optimal in their respective settings [47, 67].

3.5.2 $f^* \notin \mathcal{H}$ — With approximation error

When $f^* \notin \mathcal{H}$, we may not have $\text{Approx}_i(\overline{\lambda}) = 0$ for any $\overline{\lambda} > 0$. At $\overline{\lambda} = 0$, though we will always have $\text{Approx}_i(\overline{\lambda}) = 0$, $f_{i,\overline{\lambda}}$ may not be bounded (in other words, no element in $\mathcal{H}$ would achieve this approximation). One case where we still have $\text{Approx}_i(\overline{\lambda}) = 0$ at $\overline{\lambda} = 0$, while having $f_{i,\overline{\lambda}}$ to be bounded, is if f^* is a piece-wise kernel function over our chosen partitions i.e. $f^*(x) = f_i^*(x)$ if $x \in C_i$, with $f_i^* \in \mathcal{H}$. The bounds here would then be analogous to the previous section. In general, however, without enforcing further assumptions on f^*, it is hard to give meaningful bounds on $\text{Approx}_i(\cdot)$. While we can still proceed to obtain expressions for the *regularization, bias* and *variance* terms for $\mathbb{E}_D\left[\text{Err}(\hat{f}_C)\right]$, in this scenario it may be more instructive to compare our bounds with the bounds for the averaging estimator in [67]. Let us denote this estimator as $\hat{f}_{avg}$. To compute $\hat{f}_{avg}$, the n samples are randomly split into m groups, and $\hat{f}_{avg}$ is simply the average of the KRR estimates over all groups. In this case, we have from [67] (for any $\overline{\lambda} \in [0, \lambda]$):

$$\mathbb{E}_D\left[\text{Err}(\hat{f}_{avg})\right] \leq 2\left(\text{Approx}(\overline{\lambda}) + \mathcal{E}(n, m, \lambda, \overline{\lambda})\right) \tag{3.18}$$

where $\text{Approx}(\overline{\lambda})$ corresponds to the overall approximation error and $\mathcal{E}(R, n, m, \lambda)$ is the residual error. In particular, $\text{Approx}(\overline{\lambda}) = \mathbb{E}\left[(f^*(x) - f_{\overline{\lambda}}(x))^2\right]$ with $f_{\overline{\lambda}}$ being the overall *population* KRR estimate:

$$f_{\overline{\lambda}} = \underset{f \in \mathcal{H}}{\arg\min}\, \mathbb{E}\left[(f^*(x) - f(x))^2\right] + \overline{\lambda}\, \|f\|_{\mathcal{H}}^2 \tag{3.19}$$

Also, [67] establish the scaling: $\mathcal{E}(N, m, \lambda, \overline{\lambda}) = O\left(\lambda \|f_{\overline{\lambda}}\|_{\mathcal{H}}^2 + \frac{S(\lambda)}{n}\right)$

In contrast, for our DC-estimator, with a (potentially) different $\overline{\lambda}_i$ for each $i \in [m]$, we get (similar to Eq. (3.15)):

$$\mathbb{E}_D\left[\mathrm{Err}(\hat{f}_C)\right] \leq 2\left(\sum_{i=1}^{m} \mathrm{Approx}_i(\overline{\lambda}_i) + \mathcal{E}_C\right) \tag{3.20}$$

where we let $\mathcal{E}_C = 2\left(\sum_{i=1}^m \mathrm{Reg}_i(\overline{\lambda}_i, \lambda) + \sum_{i=1}^m \mathrm{Bias}_i(\lambda, n) + \sum_{i=1}^m \mathbb{E}_D\left[\mathrm{Var}_i(\lambda, D)\right]\right)$.

Before comparing Eq. 3.18 with Eq. 3.20, we require an additional definition. For any partition C_i, $i \in [m]$, let us define: $\mathrm{ApproxError}_i(f_{\overline{\lambda}}) = \mathbb{E}\left[(f^*(x) - f_{\overline{\lambda}}(x))^2 \mathbb{1}\left(x \in C_i\right)\right]$, i.e. the error incurred by the global estimate $f_{\overline{\lambda}}$ (Eq. (3.19)) in the i^{th} partition. Note that, $\sum_{i=1}^m \mathrm{ApproxError}_i(f_{\overline{\lambda}}) = \mathrm{Approx}(\overline{\lambda})$. To avoid confusion, we emphasize the distinction between $\mathrm{ApproxError}_i(f_{\overline{\lambda}})$ and $\mathrm{Approx}_i(\overline{\lambda}_i)$ (from Definition 2). While the former is the *local* error (in the i^{th} partition) incurred by the solution of a *global* KRR problem with regularization $\overline{\lambda}$, the latter is the *local* error incurred by the solution of a *local* KRR problem with regularization $\overline{\lambda}_i$ (as defined in Eq (3.10)).

We now have the following theorem:

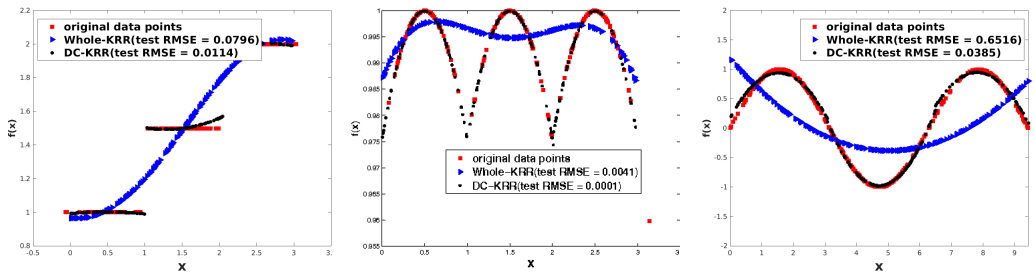
Theorem 3.3. *Consider any $\overline{\lambda} > 0$. Let $f_{\overline{\lambda}}$ be the solution of Eq. (3.19). Then, $\exists \overline{\lambda}_1, \ldots, \overline{\lambda}_m$ with $\overline{\lambda}_i \in [0, \overline{\lambda}], i \in [m]$, such that,*

$$\mathit{Approx}_i(\overline{\lambda}_i) \leq \mathit{ApproxError}_i(f_{\overline{\lambda}}) \text{ and, } \left\|f_{i,\overline{\lambda}_i}\right\|_{\mathcal{H}} = O\left(\|f_{\overline{\lambda}}\|_{\mathcal{H}} + \sqrt{\frac{\mathit{ApproxError}_i(f_{\overline{\lambda}})}{(\overline{\lambda} p_i)}}\right) \tag{3.21}$$

Thus: $\sum_{i=1}^m \mathit{Approx}_i(\overline{\lambda}_i) \leq \mathit{Approx}(\overline{\lambda})$. Moreover, if $\mathit{Approx}(\overline{\lambda}) = O\left(\overline{\lambda}\,\|f_{\overline{\lambda}}\|_{\mathcal{H}}^2\right)$, and Assumption 3.1 holds, Assumption 3.2 holds ($\forall \overline{\lambda}_i$) and Assumption 3.3 holds then: $\mathcal{E}_C = O\left(\lambda\,\|f_{\overline{\lambda}}\|_{\mathcal{H}}^2 + \frac{\sigma^2 S(\lambda)}{n}\right)$

The above theorem shows that the **approximation error term** in each partition of our estimator in $\mathbb{E}_D\left[\mathrm{Err}(\hat{f}_C)\right]$ is lower than its counterpart in $\mathbb{E}_D\left[\mathrm{Err}(\hat{f}_{avg})\right]$. Consequently, the overall approximation term is also lower. On the other hand, under suitable restrictions, the residual **estimation error** terms can be of the same order. Intuitively, this makes sense since by partitioning the space, we are fitting piece-wise kernel functions, as opposed to just a single kernel function in the averaging case. We demonstrate this through experiments in the next section.

3.6 Experiments

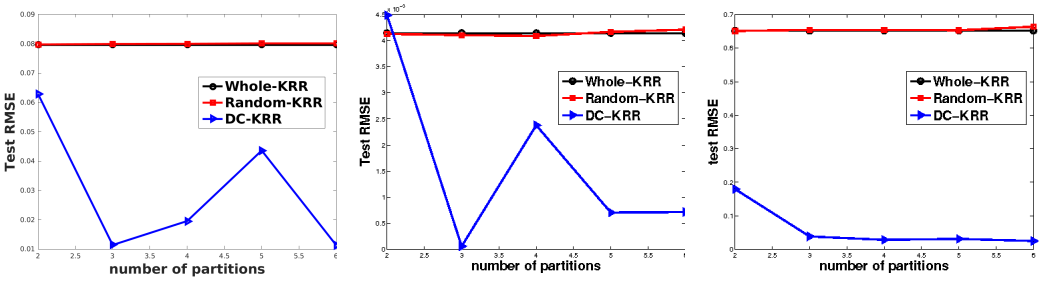


(a) Piece-wise constant (b) Piece-wise Gaussian (c) Sine function

Figure 3.1: Plots of functions obtained via Whole-KRR and DC-KRR (with 3 partitions)

In this section, we present experimental results of our proposed method on both real and toy data sets. For comparison, we tested our *DC*-estimator (DC-KRR) against the random splitting approach of [67](Random-KRR), and Kernel Ridge Regression on the entire training set (Whole-KRR).

Toy Data sets: We performed experiments on 3 toy data sets in



(a) Piece-wise constant (b) Piece-wise Gaussian (c) Sine function

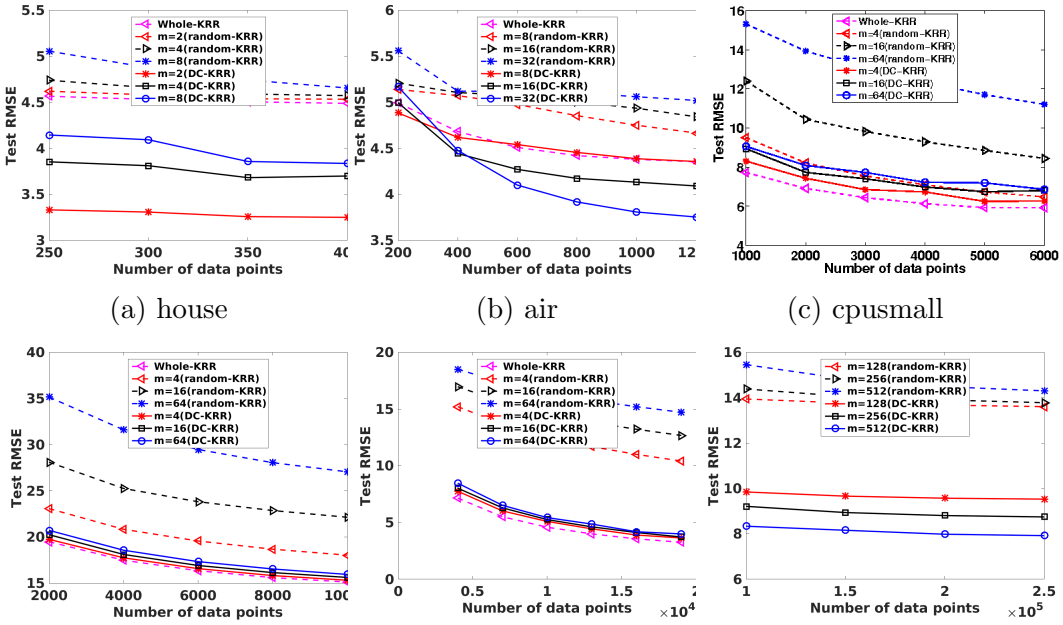
Figure 3.2: Plots of Test RMSE vs. Number of partitions on Three Toy data sets

Table 3.1: Data set statistics for real data sets used in our experiments. γ was chosen using cross-validation on the entire data set, or a sub-sample of size $10,000$ for larger data sets.

Data set	# training samples	# testing samples	# features	γ
house	404	102	13	10^{-4}
air	1,202	301	5	10^{-3}
cpusmall	6,553	1,639	12	10^{-1}
Pole	12,000	3,000	26	1
CT Slice	42,800	10,700	385	10^{-2}
Road	347,899	86,974	3	0.1

Fig 3.1. In each case, the covariate x was generated from a mixture of 3 Gaussians: $x \sim \frac{1}{3}\mathcal{N}(\mu_1, \sigma) + \frac{1}{3}\mathcal{N}(\mu_2, \sigma) + \frac{1}{3}\mathcal{N}(\mu_3, \sigma)$. For the first two toy examples, $(\mu_1 = 0.5, \mu_2 = 1.5, \mu_3 = 2.5)$ and $\sigma = 0.2$, and for the third one, $(\mu_1 = \pi/2, \mu_2 = 3\pi/2, \mu_3 = 3\pi)$ and $\sigma = 1$. The response y is $y = f^*(x) + \eta$, for different choices of f^*, and with $\eta \sim \mathcal{N}(0, 0.05)$. For each data set, we generated a training set of size 600, and a test set of size 100.

We chose f^* as: (i) a piece-wise constant function, $f^*(x) = \mathbb{1}(x \leq 1) +$



(a) house (b) air (c) cpusmall

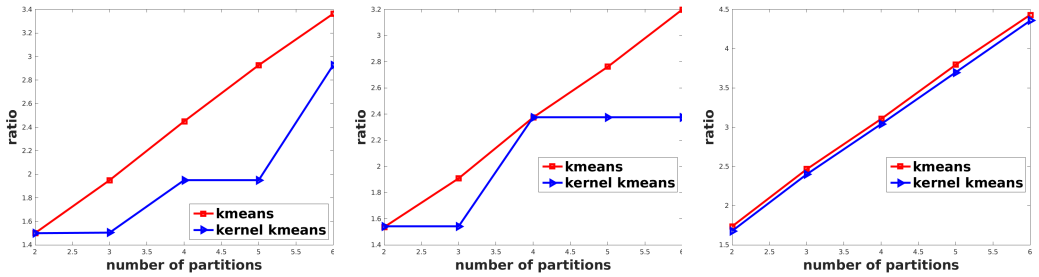
(d) Pole (e) CT slice (f) Road

Figure 3.3: Test error vs. training size on real data. m is the number of partitions, and DC-KRR uses k-means clustering. n is the number of training data points, and d is their dimension

$1.5 \times \mathbb{1}(1 < x < 2) + 2 \times \mathbb{1}(x \geq 2)$, in Fig 3.1a, (ii) a piece-wise Gaussian kernel function, $f^*(x) = \exp(-\gamma(x-0.5)^2) \times \mathbb{1}(x \leq 1) + \exp(-\gamma(x-1.5)^2) \times \mathbb{1}(1 < x < 2) + \exp(-\gamma(x-2.5)^2) \times \mathbb{1}(x \geq 2)$, with $\gamma = 0.1$, in Fig 3.1b, and (iii) a sine function, $f^*(x) = \sin(x)$, in Fig 3.1c. To obtain KRR estimate, we used a Gaussian kernel ($K(x,y) = \exp(-\gamma(x-y)^2)$) with $\gamma = 0.1$ for the first two toy data sets, and degree 2 polynomial kernel ($K(x,y) = (1+xy)^2$) for the third one. When running DC-KRR, we obtained the partition of the data points using k-means. A regularization penalty of $\lambda = 1/n$ was used, where $n =$ Total number of training points.

Fig 3.1 shows a comparison of the functions obtained using DC-KRR (run with 3 partitions) and Whole-KRR. We see that DC-KRR could approximate the true underlying function better than Whole-KRR, while still being computationally more efficient. For Fig 3.2, we varied the number of partitions, and plotted the Test-RMSE for DC-KRR, Whole-KRR and Random-KRR on toy data sets. We observe that while Random-KRR had a similar performance to Whole-KRR, DC-KRR achieved lower error than both. This is due to the lower approximation error of piece-wise estimates.

Real Data sets: We performed experiments on 6 real data sets from the UCI repository [36]. Data sets statistics are presented in Table 3.1. The data was normalized to have standard deviation 1. In all cases, we utilized a Gaussian kernel with kernel parameter γ chosen using cross-validation, as shown in Table 3.1. We varied the number of partitions, m, and the number of training points, n. When running DC-KRR, the partitions were determined using clustering, and we tested with k-means and Kernel k-means. Kernel k-means was run on a sub-sampled set of points for larger data sets. The



(a) Piece-wise constant (b) Piece-wise Gaussian (c) Sine

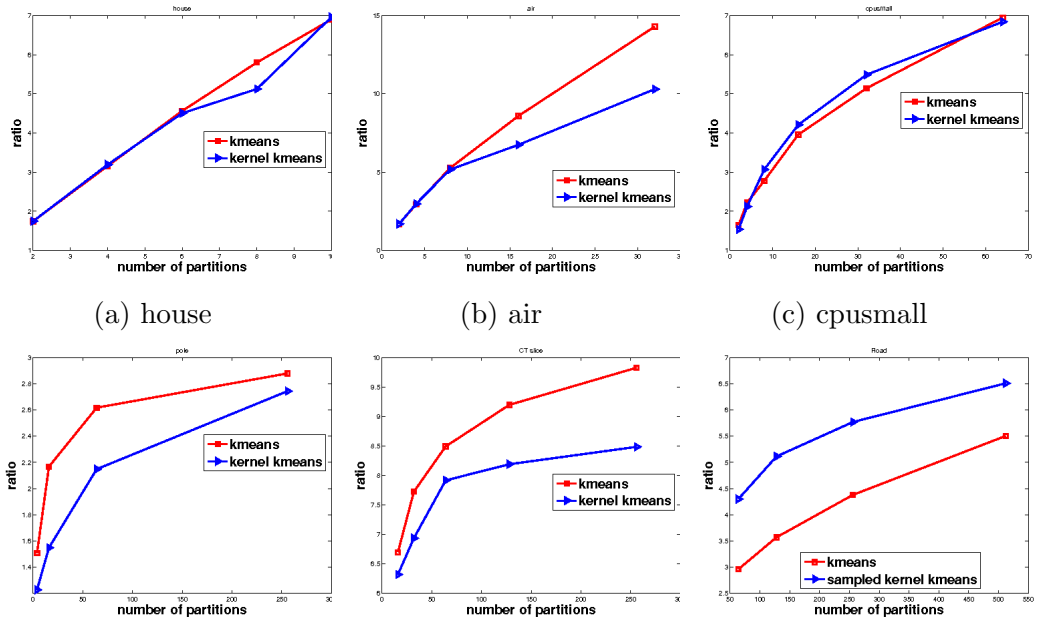
Figure 3.4: Plots of $g(\lambda)$ vs. Number of partitions on synthetic data sets

regularization penalty for KRR was chosen as $\lambda = 1/n$. The results of these experiments are presented in Table 3.2 and Fig 3.3.

In all cases, DC-KRR achieved lower test error than Random-KRR, while being comparable to Whole-KRR. Moreover, the training time for DC-KRR, when running via k-means, was similar to Random-KRR (due to the small overhead of clustering), but much faster than Whole-KRR. Interestingly, in two cases (Fig 3.3a and Fig3.3b), we found that DC-KRR also achieved lower test error than Whole-KRR. This may be a consequence of lower approximation error due to piece-wise estimates, as also alluded to earlier.

Testing Goodness of Partitioning: We also estimated $g(\lambda)$ (defined for Assumption 3.3) vs. a varying number of partitions, on both our real and toy data sets (shown in Fig 3.4 and Fig 3.5 respectively) to verify the validity of Assumption 3.3.

To estimate $S(\lambda)$ and $S_i(\lambda p_i)$, $i \in [m]$ (which comprise $g(\lambda)$), we used an SVD to compute the eigenvalues of the kernel matrix on the training samples (respectively, the kernel matrix of the training samples in partition i) and normalized this with n, the training size (respectively, n_i, the training size in partition i). In case of larger data sets, we did this on a sub-sampled version of the data set. It is known that the eigenvalues of K_D/n, with K_D being the kernel matrix on randomly sampled points D, converge to the eigenvalues of the covariance in the associated RKHS [50]. Finally, we set $\lambda = 1/n$, the same as in our earlier experiments, with n = total training size/sub-sample size.



(a) house (b) air (c) cpusmall

(d) Pole (e) CT Slice (f) Road

Figure 3.5: Plots of $g(\lambda)$ vs. number of partitions on real data sets

On real data sets, we found that while $g(\lambda)$ increases as the number of partitions increases, it continues to be a constant even for a large number of partitions in several cases, thereby justifying Assumption 3.3. On synthetic data sets, it seemed to grow at a somewhat faster rate. However, this could be attributed to lesser clustering structure, since the true number of clusters was only 3 — at which point $g(\lambda)$ is still a small constant.

Comparison with [23]: We also performed additional empirical comparisons between the approach in [23] (denoted as VP-KRR), DC-KRR (with k-means and kernel k-means) and Random-KRR, on the cpusmall data set (see Table 3.1). The main algorithmic difference between DC-KRR and VP-

KRR is that the latter proposes to obtain bounded partitions using a Voronoi partitioning of the input space, while in DC-KRR we use a clustering algorithm to obtain the partitions. The results of our tests are shown in Table3.3. We see that DC-KRR(with kernel k-means) is slightly better than VP-KRR in terms of Test RMSE, but also DC-KRR requires much lesser training time than VP-KRR. A reason for this is that Voronoi partitioning tends to produce a very unbalanced clustering. For example, when using Voronoi partitioning to generate 9 clusters, we found that the first cluster had 6484 data points out of total 6553 data points in the dataset, and the remaining clusters had very few data points. Consequently, the training time for the one cluster was almost as huge as the time it would take to train Whole-KRR.

Table 3.2: Test RMSE and Training Time on real data sets used in our experiments. # partitions is only applicable to the Random-KRR and DC-KRR columns.

Data set	# partitions	Whole-KRR		Random-KRR		DC-KRR(kernel k-means)		DC-KRR(k-means)	
		Test RMSE	Time(s)	Test RMSE	Time(s)	Test RMSE	Time(s)	Test RMSE	Time(s)
house	4	4.4822	0.08	4.5609	0.02	**3.3849**	0.18	3.8244	0.06
air	8	4.3537	2.46	4.6604	0.07	**4.2577**	0.79	4.4782	0.23
cpusmall	8	5.8853	118.98	7.1757	4.04	**5.7947**	30.86	6.4616	7.86
Pole	16	**14.7256**	1088.9	21.5768	6.15	15.0005	277.80	15.1167	11.88
CT Slice	32	**2.1165**	3840.7	10.0318	43.81	3.6100	405.38	2.4302	64.06
Road	256	-	-	13.6444	43.48	11.0550	1081.3	**8.6358**	78.16

Table 3.3: Test RMSE and Training Times on cpusmall for VP-KRR([23]), Random-KRR and DC-KRR(with k-means and kernel k-means). # partitions is only applicable to the Random-KRR and DC-KRR columns. For VP-KRR([23]), we choose the radius for obtaining voronoi partitions, r, to be α times the maximum distance between any two points in the data set, with α chosen as 0.01, 0.04, 0.07 and 0.12. After we know the number of partitions for a specific r, we generate the same number of partitions using k-means and kernel k-means (for DC-KRR), and random partitioning (for Random-KRR).

# partitions	6		9		13		40	
	Test RMSE	Time(s)	Test RMSE	Time(s)	Test RMSE	Time(s)	Test RMSE	Time(s)
VP-KRR	5.8914	129.9600	5.8653	119.9500	6.1331	113.0400	6.3026	49.69
Random-KRR	6.6232	4.49	7.3143	2.2400	7.9986	1.2100	10.1980	0.2500
DC-KRR(k-means)	6.4246	24.72	6.4610	8.6800	6.6415	4.1700	7.2206	0.9400
DC-KRR(kernel k-means)	5.7819	17.0900	5.8338	14.4700	5.8069	13.00	6.01	12.09

Chapter 4

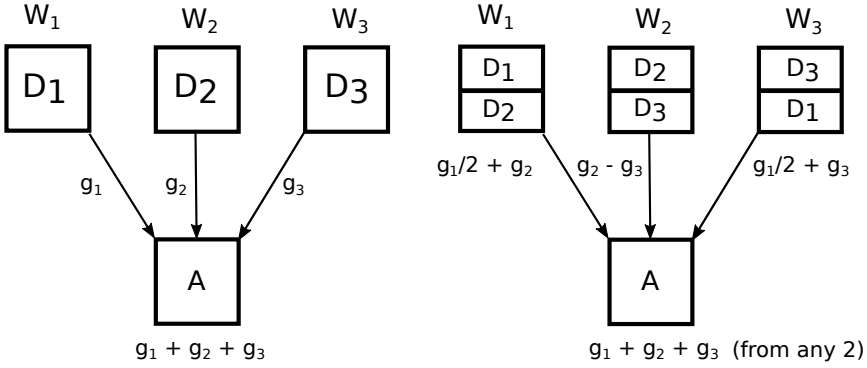
Gradient Coding: Avoiding stragglers in distributed Synchronous Gradient Descent[3]

4.1 Introduction

In this chapter, we propose a novel coding theoretic framework for mitigating stragglers in distributed learning. The central idea can be seen through the simple example of Figure 1: Consider synchronous Gradient Descent (GD) on three workers (W_1,W_2,W_3). The baseline vanilla system is shown in the left figure and operates as follows: The three workers have different partitions of the labeled data stored locally (D_1,D_2,D_3) and all share the current model. Worker 1 computes the gradient of the model on examples in partition D_1, denoted by g_1. Similarly, Workers 2 and 3 compute g_2 and g_3. The three gradient vectors are then communicated to a central node (called the master/aggregator) A which computes the full gradient by summing these vectors $g_1 + g_2 + g_3$ and updates the model with a gradient step. The new model is then sent to the workers and the system moves to the next round (where

[3]This chapter is based on [57]. The author of this work was the first author and primary contributor to [57]. The author collaborated on proposing a linear algebra view of the straggler problem considered here. Thereafter, the author proposed an algorithm using coding theoretic tools to solve it. The author also collaborated on performing experiments to validate the efficacy of the proposed solution, as well as writing the paper.

the same examples or other labeled examples, say D_4,D_5,D_6, will be used in the same way). The problem is that sometimes worker nodes can be strag-



(a) Naive synchronous gradient descent

(b) Gradient coding: The vector $g_1 + g_2 + g_3$ is in the span of *any two* out of the vectors $g_1/2 + g2$, $g_2 - g_3$ and $g_1/2 + g_3$.

Figure 4.1: The idea of Gradient Coding.

glers [18, 27, 32] *i.e.* delay significantly in computing and communicating gradient vectors to the master. This is especially pronounced for cheaper virtual machines in the cloud. For example on `t2.micro` machines on Amazon EC2, as can be seen in Figure 4.2: some machines can be 5× slower in computing and communicating gradients compared to typical performance.

First, we discuss one way to resolve this problem if we replicate some data across machines by considering the placement in Fig.1 (b) but without coding. As can be seen, in Fig. 1 (b) each example is replicated two times using a specific placement policy. Each worker is assigned to compute two gradients on the two examples they have for this round. For example, W_1

will compute vectors g_1 and g_2. Now let's assume that W_3 is the straggler. If we use control messages, W_1, W_2 can notify the master A that they are done. Subsequently, if feedback is used, the master can ask W_1 to send g_1 and g_2 and W_2 to send g_3. These feedback control messages can be much smaller than the actual gradient vectors but are still a system complication that can cause delays. However, feedback makes it possible for a centralized node to coordinate the workers, thereby avoiding stragglers. One can also reduce network communication further by simply asking W_1 to send the sum of two gradient vectors $g_1 + g_2$ instead of sending both. The master can then create the global gradient on this batch by summing these two vectors. Unfortunately, which linear combination must be sent depends on who is the straggler: If W_2 was the straggler then W_1 should be sending g_2 and W_3 sending $g_1 + g_3$ so that their sum is the global gradient $g_1 + g_2 + g_3$.

In this work we show that feedback and coordination is not necessary: every worker can send a *single linear combination of gradient vectors* without knowing who the straggler will be. The main coding theoretic question we investigate is how to design these linear combinations so *that any two* (or any fixed number generally) contain the $g_1 + g_2 + g_3$ vector in their span. In our example, in Fig. 4.1b, W_1 sends $\frac{1}{2}g_1+g_2$, W_2 sends g_2-g_3 and W_3 sends $\frac{1}{2}g_1+g_3$. The reader can verify that A can obtain the vector $g_1 + g_2 + g_3$ from *any two out of these three vectors*. For instance, $g_1+g_2+g_3 = 2\left(\frac{1}{2}g_1 + g_2\right) - (g_2 - g_3)$. We call this idea *gradient coding*.

We consider this problem in the general setting of n machines and **any**

s stragglers. We first establish a lower bound: to compute gradients on all the data in the presence of **any** s stragglers, each partition must be replicated $s+1$ times across machines. We propose two placement and gradient coding schemes that match this optimal $s+1$ replication factor. We further consider a partial straggler setting, wherein we assume that a straggler can compute gradients at a fraction of the speed of others, and show how our scheme can be adapted to such scenarios. All proofs for this chapter can be found in Appendix D.

We also compare our scheme with the popular *ignoring the stragglers* approach [11]: simply doing a gradient step when most workers are done. We see that while ignoring the stragglers is faster, this loses some data which can hurt the generalization error. This can be especially pronounced in supervised learning with unbalanced labels or heavily unbalanced features since a few examples may contain critical, previously unseen information.

4.1.1 The Effects of Stragglers

In Figure 4.2, we show the average time required for 50 `t2.micro` Amazon EC2 instances to communicate gradients to a single master machine (a `c3.8xlarge` instance). We observe that a few worker machines incurred a communication delay of up to $5\times$ the typical behavior. Interestingly, throughout the timescale of our experiments (a few hours), the straggling behavior was consistent in the same machines.

We have also experimented extensively with other Amazon EC2 in-

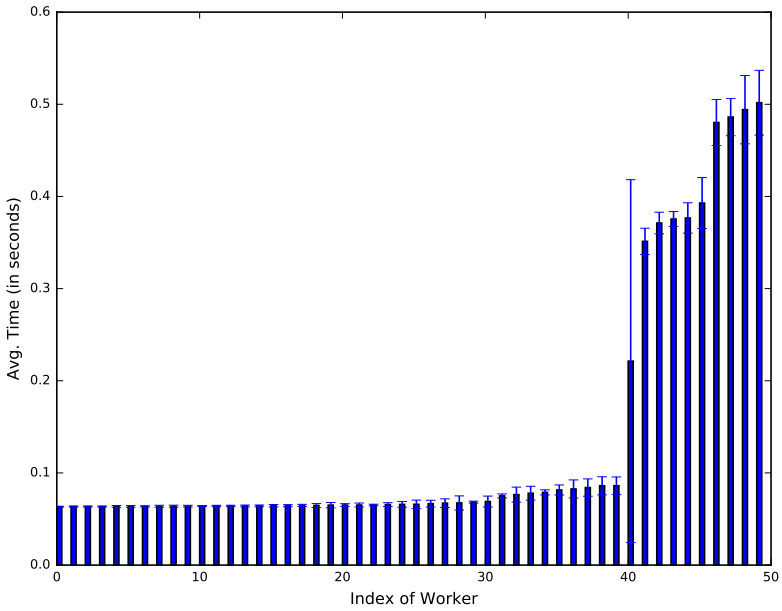


Figure 4.2: Average communication times, measure over 100 rounds, for a vector of dimension $p = 500000$ using $n = 50$ `t2.micro` worker machines (and a `c3.8xlarge` master machine). Error bars indicate one standard deviation.

stances: Our finding is that cheaper instance types have significantly higher variability in performance. This is especially true for `t2` type instance which on AWS are described as having *Burstable Performance*. Fortunately, these machines have very low cost.

The choices of the number and type of workers used in training big models ultimately depends on total cost and time needed until deployment. The main message of this work is that going for very low-cost instances and using coding to mitigate stragglers, may be a sensible choice for some learning problems.

4.1.2 Related Work

The slow machine problem is the Achilles heel of many distributed learning systems that run in modern cloud environments. Recognizing that, some recent work has advocated asynchronous approaches [27, 32, 41] to learning. While asynchronous updates are a valid way to avoid slow machines, they do give up many other desirable properties, including faster convergence rates, amenability to analysis, and ease of reproducibility and debugging.

Attacking the straggling problem in synchronous machine learning algorithms has surprisingly not received much attention in the literature. There do exist general systems solutions such as speculative execution [64] but we believe that approaches tailored to machine learning can be vastly more efficient. In [11] the authors use synchronous minibatch SGD and request a small number of additional worker machines so that they have an adequate minibatch size even when some machines are slow. However, this approach does not handle well machines that are consistently slow and the data on those machines might never participate in training. In [42] the authors describe an approach for dealing with failed machines by approximating the loss function in the failed partitions with a linear approximation at the last iterate before they failed. Since the linear approximation is only valid at a small neighborhood of the model parameters, this approach can only work if failed data partitions are restored fairly quickly.

The work of [31] is the closest in spirit to our work, using coding theory and treating stragglers as erasures in the transmission of the computed

results. However, we focus on codes for recovering the batch gradient of *any* loss function while [31] and the more recent work of [21] describe techniques for mitigating stragglers in two different distributed applications: data shuffling and matrix multiplication. We also mention [34], which investigates a generalized view of the coding ideas in [31], showing that their solution is a single operating point in a general scheme of trading off latency of computation to the load of communication. Further closely related work has shown how coding can be used for distributed MapReduce, as well as a similar communication and computation tradeoff [33, 35]. All these prior works develop novel coding techniques, but do not code across gradient vectors in the way we are proposing in our work.

4.2 Notation and Preliminaries

Given data $\mathbf{D} = \{(x_1, y_1), \ldots, (x_d, y_d)\}$, with each tuple $(x, y) \in \mathbb{R}^p \times \mathbb{R}$, several machine learning tasks aim to solve the following problem:

$$\beta^* = \arg\min_{\beta \in \mathbb{R}^p} \sum_{i=1}^{d} \ell\left(\beta; x_i, y_i\right) + \lambda R(\beta) \tag{4.1}$$

where $\ell(\cdot)$ is a task-specific loss function, and $R(\cdot)$ is a regularization function. Typically, this optimization problem can be solved using gradient-based approaches. Let $g := \sum_{i=1}^{d} \nabla \ell(\beta^{(t)}; x_i, y_i)$ be the gradient of the loss at the current model $\beta^{(t)}$. Then the updates to the model are of the form:

$$\beta^{(t+1)} = h_R\left(\beta^{(t)}, g\right) \tag{4.2}$$

where h_R is a gradient-based optimizer, which also depends on $R(\cdot)$. Several methods such as gradient descent, accelerated gradient, conditional gradient (Frank-Wolfe), proximal methods, LBFGS, and bundle methods fit in this framework. However, if the number of samples, d, is large, a computational bottleneck in the above update step is the computation of the gradient, g, whose computation can be distributed.

Notation: Throughout this chapter, we let d denote the number of samples, n denote the number of workers, k denote the number of data partitions, and s denote the number of stragglers/failures. The n workers are denoted as $W_1, W_2, \ldots, W_n$. The partial gradients over k data partitions are denoted as $g_1, g_2, \ldots, g_k$. The i^{th} row of some matrices A or B is denoted as a_i or b_i respectively. For any vector $\mathbf{x} \in \mathbb{R}^n$, $\text{supp}(\mathbf{x})$ denotes its support *i.e.* $\text{supp}(\mathbf{x}) = \{i \,|\, \mathbf{x}_i \neq 0\}$, and $\|\mathbf{x}\|_0$ denotes its ℓ_0-norm i.e. the cardinality of the support. $\mathbf{1}_{p\times q}$ and $\mathbf{0}_{p\times q}$ denote all 1s and all 0s matrices respectively, with dimension $p \times q$. Finally, for any $r \in \mathbb{N}$, $[r]$ denotes the set $\{1, \ldots, r\}$.

4.2.1 The General Setup

We can generalize the scheme in Figure 4.1b to n workers and k data partitions by setting up a system of linear equations:

$$AB = \mathbf{1}_{f\times k} \tag{4.3}$$

where f denotes the number of combinations of surviving workers/non-stragglers, $\mathbf{1}_{f\times k}$ is the all 1s matrix of dimension $f \times k$, and we have matrices $A \in \mathbb{R}^{f\times n}$,

$B \in \mathbb{R}^{n \times k}$.

We associate the i^{th} row of B, b_i, with the i^{th} worker, W_i. The support of b_i, supp(b_i), corresponds to the data partitions that worker W_i has access to, and the entries of b_i encode a linear combination over their gradients that worker W_i transmits. Let $\bar{g} \in \mathbb{R}^{k \times d}$ be a matrix with each row being the partial gradient of a data partition i.e.

$$\bar{g} = [g_1, g_2, \ldots, g_k]^T.$$

Then, worker W_i transmits $b_i\bar{g}$. Note that to transmit $b_i\bar{g}$, W_i only needs to compute the partial gradients on the partitions in supp(b_i). Now, each row of A is associated with a specific failure/straggler scenario, to which tolerance is desired. In particular, any row a_i, with support supp(a_i), corresponds to the scenario where the worker indices in supp(a_i) are alive/non-stragglers. Also, by the construction in Eq. (4.3), we have:

$$a_i B\bar{g} = [1, 1, \ldots, 1]\bar{g} = \left(\sum_{j=1}^{k} g_j\right)^T \text{ and,} \tag{4.4}$$

$$a_i B\bar{g} = \sum_{k \in \text{supp}(a_i)} a_i(k)(b_k\bar{g}) \tag{4.5}$$

where $a_i(k)$ denotes the k^{th} element of the row a_i. Thus, the entries of a_i encode a linear combination which, when taken over the transmitted gradients of the alive/non-straggler workers, $\{b_k\bar{g}\}_{k \in \text{supp}(a_i)}$, would yield the full gradient.

Going back to the example in Fig. 4.1b, the corresponding A and B

matrices under the above generalization are:

$$A = \begin{pmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & -1 & 0 \end{pmatrix}, \text{ and } B = \begin{pmatrix} 1/2 & 1 & 0 \\ 0 & 1 & -1 \\ 1/2 & 0 & 1 \end{pmatrix} \tag{4.6}$$

with $f = 3, n = 3, k = 3$. It is easy to check that $AB = \mathbf{1}_{3\times 3}$. Also, since every row of A here has exactly one zero, we say that this scheme is robust to any one straggler.

In general, we shall seek schemes, through the construction of (A, B), which are robust to **any** s stragglers.

The rest of this chapter is organized as follows. In Section 4.3 we provide two schemes applicable to any number of workers n, under the assumption that stragglers can be arbitrarily slow to the extent of total failure. In Section 4.4, we relax this assumption to the case of worker slowdown (with known slowdown factor), instead of failure, and show how our constructions can be appended to be more effective. Finally, in Section 4.5 we present results of empirical tests using our proposed distribution schemes on Amazon EC2.

4.3 Full Stragglers

In this section, we consider schemes robust to any s stragglers, given n workers (with $s < n$). We assume that any straggler is (what we call) a *full straggler i.e.* it can be arbitrarily slow to the extent of complete failure. We show how to construct the matrices A and B, with $AB = \mathbf{1}$, such that the scheme (A, B) is robust to **any** s full stragglers.

Consider any such scheme (A, B). Since every row of A represents a set of non-straggler workers, all possible sets over $[n]$ of size $(n-s)$ must be supports in the rows of A. Thus $f = \binom{n}{n-s} = \binom{n}{s}$ *i.e.* the total number of failure scenarios is the number of ways to choose s stragglers out of n workers. Now, since each row of A represents a linear span over some rows of B, and since we require $AB = \mathbf{1}$, this leads us to the following condition on B:

Condition 1 (B-Span)**.** *Consider any scheme (A, B) robust to **any** s stragglers, given n workers (with $s < n$). Then we require that for every subset $I \subseteq [n], |I| = n - s$:*

$$\mathbf{1}_{1\times k} \in span\{b_i \,|\, i \in I\} \tag{4.7}$$

where $span\{\cdot\}$ is the span of vectors.

The B-Span condition above ensures that the all **1**s vector lies in the span of any $n - s$ rows of B. This is of course necessary. However, it is also sufficient. In particular, given a B satisfying Condition 1, we can construct A such that $AB = \mathbf{1}$, and A has the support structure discussed above. The construction of A is described in Algorithm 4.1 (in MATLAB syntax), and we have the following lemma.

Lemma 4.1. *Consider $B \in \mathbb{R}^{n\times k}$ satisfying Condition 1 for some $s < n$. Then, Algorithm 4.1, with input B and s, yields an $A \in \mathbb{R}^{\binom{n}{s}\times n}$ such that $AB = \mathbf{1}_{\binom{n}{s}\times n}$ and the scheme (A, B) is robust to any s full stragglers.*

Based on Lemma 4.1, to obtain a scheme (A, B) robust to **any** s stragglers, we only need to furnish a B satisfying Condition 1. A trivial B that

Algorithm 4.1 Algorithm to compute A

Input : B satisfying Condition 1, $s(< n)$
Output: A such that $AB = \mathbf{1}_{\binom{n}{s} \times n}$

$f = \text{binom}(n, s)$;
$A = \text{zeros}(f, n)$;
foreach $I \subseteq [n]$ *s.t.* $|I| = (n - s)$ **do**
  $a = \text{zeros}(1, k)$;
  $x = \text{ones}(1, k)/B(I, :)$;
  $a(I) = x$;
  $A = [A; a]$;
end

works is $B = \mathbf{1}_{n \times k}$, the all ones matrix. However, this is wasteful since it implies that each worker gets all the partitions and computes the full gradient. Our goal is to construct B satisfying Condition 1 while also being as sparse as possible in each row. In this regard, we have the following theorem, which gives a lower bound on the number of non-zeros in any row of B.

Theorem 4.1 (Lower Bound on B's density)**.** *Consider any scheme* (A, B) *robust to* ***any*** s *stragglers, given* n *workers (with* $s < n$*) and* k *partitions. Then, if all rows of* B *have the same number of non-zeros, we must have:* $\|b_i\|_0 \geq \frac{k}{n}(s + 1)$ *for any* $i \in [n]$.

Theorem 4.1 implies that any scheme (A, B) that assigns the same amount of data to all the workers must assign at least $\frac{s+1}{n}$ fraction of the data to each worker. Since this fraction is independent of k, for the remainder of this chapter we shall assume that $k = n$ i.e. the number of partitions is the same as the number of workers. In this case, we want B to be a square matrix

satisfying Condition 1, with each row having at least $(s+1)$ non-zeros. In the sequel, we demonstrate two constructions for B which satisfy Condition 1 and achieve the density lower bound.

4.3.1 Fractional Repetition Scheme

In this section, we provide a construction for B that works by replicating the task done by a subset of the workers. We note that this construction is only applicable when the number of workers, n, is a multiple of $(s+1)$, where s is the number of stragglers we seek tolerance to. In this case, the construction is as follows:

- We divide the n workers into $(s+1)$ groups of size $(n/(s+1))$.
- In each group, we divide all the data equally and disjointly, assigning $(s+1)$ partitions to each worker
- All the groups are replicas of each other
- When finished computing, every worker transmits the sum of its partial gradients

Fig. 4.3 shows an instance of the above construction for $n = 6, s = 2$. A general description of B constructed in this way (denoted as B_{frac}) is shown in Eq. (4.9). Each group of workers in this scheme can be denoted by a block

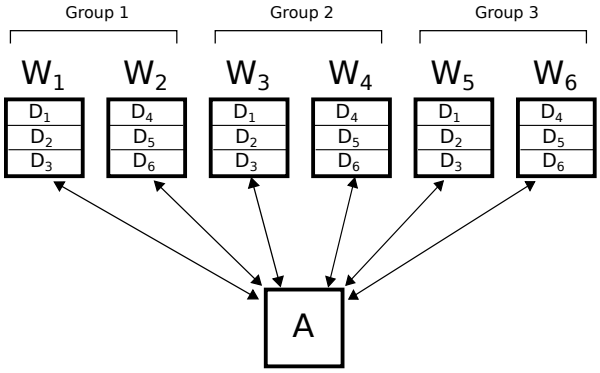


Figure 4.3: Fractional Repetition Scheme for $n = 6, s = 2$

matrix $\overline{B}_{\text{block}}(n,s) \in \mathbb{R}^{\frac{n}{s+1} \times n}$. We define:

$$\overline{B}_{\text{block}}(n,s) = \begin{bmatrix} \mathbf{1}_{1\times(s+1)} & \mathbf{0}_{1\times(s+1)} & \cdots & \mathbf{0}_{1\times(s+1)} \\ \mathbf{0}_{1\times(s+1)} & \mathbf{1}_{1\times(s+1)} & \cdots & \mathbf{0}_{1\times(s+1)} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times(s+1)} & \mathbf{0}_{1\times(s+1)} & \cdots & \mathbf{1}_{1\times(s+1)} \end{bmatrix}_{\frac{n}{s+1}\times n} \tag{4.8}$$

Thus, the first worker in the group gets the first $(s+1)$ partitions, the second worker gets the second $(s+1)$ partitions, and so on. Then, B is simply $(s+1)$ replicated copies of $\overline{B}_{\text{block}}(n,s)$:

$$B = B_{frac} = \begin{bmatrix} \overline{B}^{(1)}_{\text{block}} \\ \overline{B}^{(2)}_{\text{block}} \\ \vdots \\ \overline{B}^{(s+1)}_{\text{block}} \end{bmatrix}_{n\times n} \tag{4.9}$$

where for each $t \in \{1, \ldots, s+1\}$, $\overline{B}^{(t)}_{\text{block}} = \overline{B}_{\text{block}}(n,s)$.

It is easy to see that this construction can yield robustness to **any** s stragglers. Since any particular partition of data is replicated over $(s+1)$

workers, any s stragglers would leave at least one non-straggler worker to process it. We have the following theorem.

Theorem 4.2. *Consider B_{frac} constructed as in Eq.* (4.9)*, for a given number of workers n and stragglers $s(< n)$. Then, B_{frac} satisfies the B-Span condition (Condition 1). Consequently, the scheme (A, B_{frac}), with A constructed using Algorithm 4.1, is robust to any s stragglers.*

The construction of B_{frac} matches the density lower bound in Theorem 4.1 and, the above theorem shows that the scheme (A, B_{frac}), with A constructed from Algorithm 4.1, is robust to s stragglers.

4.3.2 Cyclic Repetition Scheme

In this section we provide an alternate construction for B which also matches the lower bound in Theorem 4.1 and satisfies Condition 1. However, in contrast to construction in the previous section, this construction does not require n to be divisible by $(s + 1)$. Here, instead of assigning disjoint collections of partitions, we consider a cyclic assignment of $(s + 1)$ partitions to the workers. We construct a $B = B_{cyc}$ with the following support structure:

$$\text{supp}(B_{cyc}) = \begin{bmatrix} \overbrace{\star & \star & \cdots & \star & \star}^{\text{s+1}} & 0 & 0 & \cdots & 0 & 0 \\ 0 & \star & \star & \cdots & \star & \star & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \star & \star & \cdots & \star & \star \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ \star & \cdots & \star & \star & 0 & 0 & \cdots & 0 & 0 & \star \end{bmatrix}_{n \times n} \tag{4.10}$$

where $\star$ indicates non-zero entries in B_{cyc}. So, the first row of B_{cyc} has its first $(s+1)$ entries assigned as non-zero. As we move down the rows, the positions of the $(s+1)$ non-zero entries shift one step to the right, and cycle around until the last row.

Given the support structure in Eq. 4.10, the actual non-zero entries must be carefully assigned in order to satisfy Condition 1. The basic idea is to pick every row of B_{cyc}, with its particular support, to lie in a suitable subspace S that contains the all ones vector $\mathbf{1}_{n\times 1}$. We consider a $(n-s)$ dimensional subspace, $S = \{x \in \mathbb{R}^n \,|\, Hx = 0,\, H \in \mathbb{R}^{s\times n}\}$ *i.e.* the null space of the matrix H, for some H satisfying $H\mathbf{1} = 0$. Now, to make the rows of B_{cyc} lie in S, we require that the null space of H must contain vectors with all the different supports in Eq. 4.10. This turns out to be equivalent to requiring that any s columns of H are linearly independent, and is also referred to as the MDS property in coding theory. We show that a random choice of H suffices for this, and we are able to construct a B_{cyc} with the support structure in Eq. 4.10. Moreover, for any $(n-s)$ rows of B_{cyc}, we

show that their linear span also contains $\mathbf{1}_{n\times 1}$, thereby satisfying Condition 1. Algorithm 4.2 describes the construction of B_{cyc} (in MATLAB syntax) and, we have the following theorem.

Algorithm 4.2 Algorithm to construct $B = B_{cyc}$

Input : n, $s(< n)$
Output: $B \in \mathbb{R}^{n\times n}$ with $(s+1)$ non-zeros in each row

$H = \text{randn}(s, n)$;
$H(:, n) = -\text{sum}(H(:, 1 : n-1), 2)$;
$B = \text{zeros}(n)$;
for $i = 1 : n$ **do**
  $j = \text{mod}(i - 1 : s + i - 1, n) + 1$;
  $B(i, j) = [1; -H(:, j(2 : s + 1))\backslash H(:, j(1))]$;
end

Theorem 4.3. *Consider B_{cyc} constructed using the randomized construction in Algorithm 4.2, for a given number of workers n and stragglers $s(< n)$. Then, with probability 1, B_{cyc} satisfies the B-Span condition (Condition 1). Consequently, the scheme (A, B_{cyc}), with A constructed using Algorithm 4.1, is robust to any s stragglers.*

4.4 Partial Stragglers

In this section, we revisit our earlier assumption of *full stragglers*. Under a *full straggler* assumption, Theorem 4.1 shows that any non-straggler worker must incur an $(s + 1)$-factor overhead in computation, if we want to attain tolerance to any s stragglers. This may be prohibitively huge in many situations. One way to mitigate this is by allowing at least some work to be

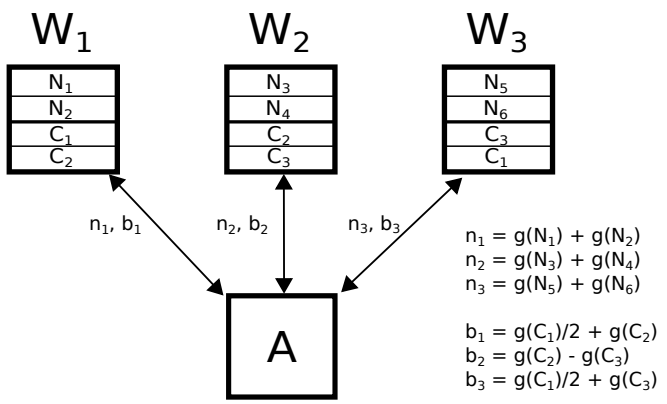


Figure 4.4: Scheme for Partial Stragglers, $n = 3, s = 1, \alpha = 2$. $g(\cdot)$ represents the partial gradient.

done also by the straggling workers. Therefore, in this section, we consider a more plausible scenario of slow workers, but assume a known slowdown factor. We say that a straggler is an *α-partial straggler* (with $\alpha > 1$) if it is at most α slower than any non-straggler. This means that if a non-straggler completes a task in time T, an *α-partial straggler* would require at most αT time to complete it. Now, we augment our previous schemes (in Section 4.3.1 and Section 4.3.2) to be robust to **any** s stragglers, assuming that any straggler is an *α-partial straggler*.

Note that our earlier constructions are still applicable: a scheme (A, B), with $B = B_{frac}$ or $B = B_{cyc}$, would still provide robustness to s partial stragglers. However, given that no machine is slower than a factor of α, a more efficient scheme is possible by exploiting at least some computation on every machine. Our basic idea is to couple our earlier schemes with a naive distribution scheme, but on different parts of the data. We split the data into a *naive* component, and a *coded* component. The key is to do the split such

that whenever an α-partial straggler is done processing its *naive* partitions, a non-straggler would be done processing both its *naive* and *coded* partitions.

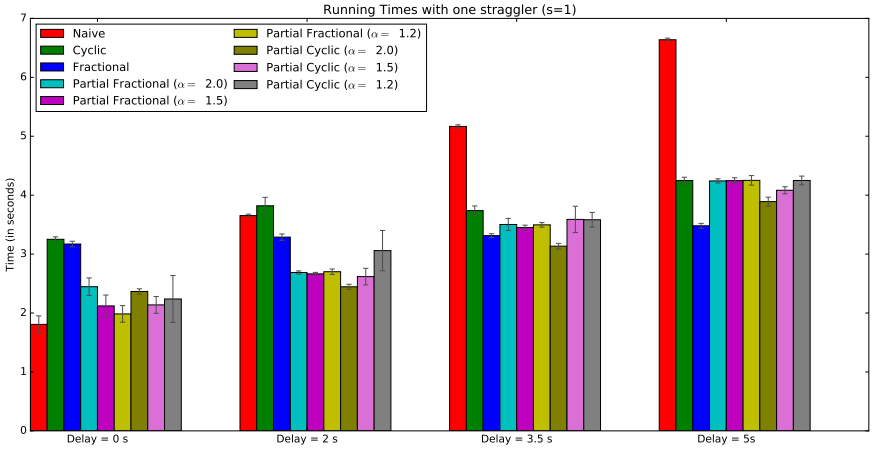
In general, for any (n, s, α), our two-stage scheme works as follows:

- We split the data $\mathbf{D}$ into $n + n\frac{s+1}{\alpha-1}$ equal-sized partitions — of which n partitions are *coded* components, and the rest are *naive* components
- Each worker gets $\frac{s+1}{\alpha-1}$ *naive* partitions, distributed disjointly.
- Each worker gets $(s + 1)$ *coded* partitions, distributed according to an (A, B) distribution scheme robust to s stragglers (e.g. with $B = B_{frac}$ or $B = B_{cyc}$)
- Any worker, W_i, first processes all its *naive* partitions and sends the sum of their gradients to the aggregator. It then processes its *coded* partitions, and sends a linear combination, as per the (A, B) distribution scheme.

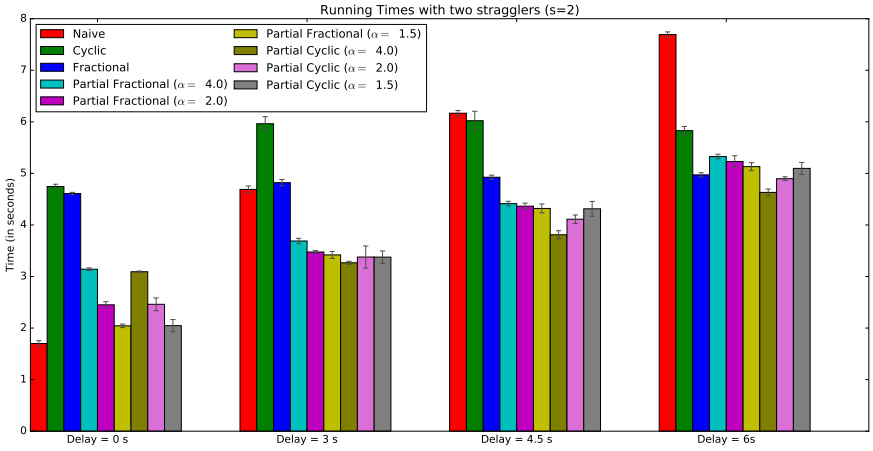
Note that each worker now has to send two partial gradients (instead of one, as in earlier schemes). However, a speedup gained in processing a smaller fraction of the data may mitigate this overhead in communication, since each non-straggler only has to process a $\frac{s+1}{n}\left(\frac{\alpha}{s+\alpha}\right)$ fraction of the data, as opposed to a $\frac{s+1}{n}$ fraction in *full straggler* schemes. Thus, when computation is the bottleneck, adopting a partial stragglers scheme may not hurt the overall efficiency. On the other hand, when communication is the bottleneck (and if a $2\times$ overhead is prohibitive), a full straggler scheme may be a better choice even with its (s+1)-factor overhead in computation for the non-straggler workers.

Fig. 4.4 illustrates our two-stage strategy for $n = 3, s = 1, \alpha = 2$. We see that each non-straggler gets $4/9 = 0.44$ fraction of the data, instead of a $2/3 = 0.67$ fraction (for e.g. in Fig 4.1b).

4.5 Experiments



(a) $s = 1$ Straggler



(b) $s = 2$ Stragglers

Figure 4.5: Empirical running times on Amazon EC2 with $n = 12$ machines for $s = 1$ and $s = 2$ stragglers. In this experiment, the stragglers are artificially delayed while the other machines run at normal speed. We note that the partial straggler schemes have much lower data replication, for example with $\alpha = 1.2$ we need to only replicate approximately 10% of the data.

In this section, we present experimental results on Amazon EC2, comparing our proposed gradient coding schemes with baseline approaches. We compare our approaches against: (1) the *naive* scheme, where the data is divided uniformly across all workers without replication and the aggregator waits for all workers to send their gradients, and (2) the *ignoring s stragglers* scheme where the data is divided as in the naive scheme, however the aggregator performs an update step after any $n - s$ workers have successfully sent their gradient.

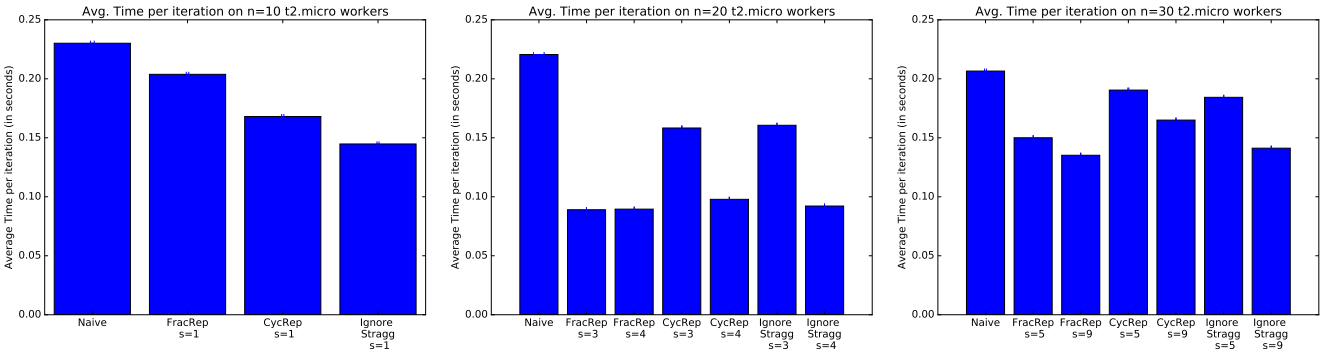


Figure 4.6: Avg. Time per iteration on Amazon Employee Access dataset.

4.5.1 Experimental setup

We implemented all methods in python using MPI4py [16], an open source MPI implementation. Based on the method being considered, each worker loads a certain number of partitions of the data into memory before starting the iterations. In iteration t the aggregator sends the latest model $\beta^{(t)}$ to all the workers (using `Isend()`). Each worker receives the model (using

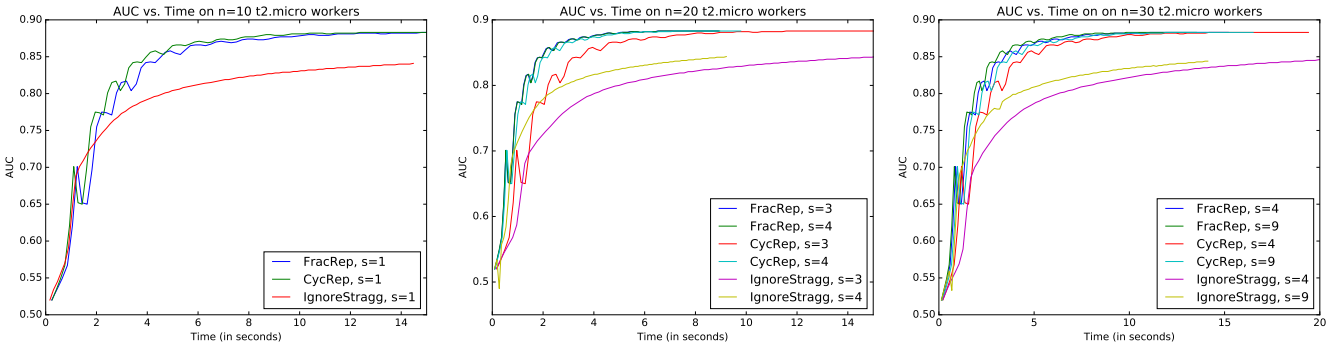


Figure 4.7: AUC vs Time on Amazon Employee Access dataset. The two proposed methods are FracRep and CycRep compared against the frequently used approach of Ignoring s stragglers. As can be seen, gradient coding achieves significantly better generalization error on a true holdout.

`Irecv()`) and starts a gradient computation. Once finished, it sends its gradient(s) back to the aggregator. When sufficiently many workers have returned with their gradients, the aggregator computes the overall gradient, performs a descent step, and moves on to the next iteration.

Our experiments were performed using two different worker instance types on Amazon EC2: `m1.small` and `t2.micro` — these are very small, very low-cost EC2 instances. We also observed that our system was often bottlenecked by the number of incoming connections *i.e.* all workers trying to talk to the master concurrently. For that reason, and to mitigate this additional overhead to some degree, we used a larger master instance of `c3.8xlarge` in our experiments.

We ran the various approaches to train logistic regression models, a well-understood convex problem that is widely used in practice. Moreover,

Logistic regression models are often expanded by including interaction terms that are often one-hot encoded for categorical features. This can lead to 100's of thousands of parameters (or more) in the trained models. To train the logistic regression models for using our proposed scheme (or the *naive* scheme), we used Nesterov's Accelerated Gradient descent with a constant learning rate, where the constant was chosen optimally from a range. Note that other optimizers such as LBFGS would have also been applicable here since we obtain the full gradient in our schemes. For the *ignoring s stragglers* approach, we used gradient descent with a learning rate of $c_1/(t + c_2)$ (which is typical for SGD), where c_1 and c_2 were also chosen optimally in a range. We did not use NAG here since it is unstable to noisy gradients. While we do not present any empirical results, we refer the reader to [20] for a theoretical and empirical analysis of the effect of noisy gradients in NAG. Thus another advantage of our schemes over *ignoring s stragglers* is that the latter cannot be combined with NAG because errors may quickly accumulate and eventually cause the method to diverge.

4.5.2 Results

Artificial Dataset: In our first experiment, we solved a logistic regression problem on a artificially generated dataset. We generated a dataset of $d = 554400$ samples $\mathbf{D} = \{(x_1, y_1), \ldots, (x_d, y_d)\}$, using the model $x \sim 0.5 \times \mathcal{N}(\mu_1, I) + 0.5 \times \mathcal{N}(\mu_2, I)$ (for random $\mu_1, \mu_2 \in \mathbb{R}^p$), and $y \sim Ber(\kappa)$, with $\kappa = 1/(\exp(2x^T\beta^*) + 1)$, where $\beta^* \in \mathbb{R}^p$ is the true regressor. In our

experiments, we used a model dimension of $p = 100$, and chose β^* randomly.

In this experiment, we also artificially added delays to s random workers in each iteration (using `time.sleep()`). Figure 4.5 presents the results of our experiments with $s = 1$ and $s = 2$ stragglers, on a cluster of $n = 12$ `m1.small` machines. As expected, the baseline *naive* scheme that waits for the stragglers has poorer performance as the delay increases. The *Cyclic* and *Fractional* schemes were designed for one straggler in Figure 4.5a and for two stragglers in Figure 4.5b. Therefore, we expect that these two schemes would not be influenced at all by the delay of the stragglers (up to some variance due to implementation overheads). The *partial straggler* schemes were designed for various α. Recall that for partial straggler schemes, α denotes the *slowdown* factor.

Real Dataset: Next, we trained a logistic regression model on the Amazon Employee Access dataset from Kaggle [1]. We used $d = 26200$ training samples, and a model dimension of $p = 241915$ (after one-hot encoding with interaction terms). These experiments were run on $n = 10, 20, 30$ `t2.micro` instances on Amazon EC2.

In Figure 4.7 we show the Generalization AUC of our method (FracRep and CycRep) versus *ignoring s stragglers* (IgnoreStragg). As can be seen, Gradient coding achieved significantly better generalization error. We emphasize that the results in figures 4.6 and 4.7 do not use any artificial straggling, only

[1] https://www.kaggle.com/c/amazon-employee-access-challenge

the natural delays introduced by the EC2 cluster.

How is this stark difference possible? When stragglers were ignored we were, at best, receiving a stochastic gradient (when random machines are straggling in each iteration). As alluded to earlier, in this case the best we could do as an optimization algorithm is to run gradient descent as it is robust to noise. When using gradient coding however, we could retrieve the full gradient which gave us access to faster optimization algorithms. In Figure 4.7 we used Nesterov's Accelerated Gradient (NAG).

Another advantage of using full gradients is that we can guarantee that we are training on the same distribution as the one the training set was drawn from. This is not true for the approach that ignores stragglers. If a particular machine is more likely to be a straggler, samples on that machine will likely be underrepresented in the final model, unless particular countermeasures are deployed. There may even be inherent reasons why a particular sample will systematically be excluded when we ignore stragglers. For example, in structured models such as linear-chain CRFs, the computation of the gradient is proportional to the length of the sequence. Therefore, extraordinarily long examples can be ignored very frequently.

Chapter 5

Conclusion

In this thesis, we have studied three problems that typically invoke low-dimensional structural assumptions, or distributed approaches, to solve them. For these problems, we have shown approaches that further account for specific structural characteristics or computational bottlenecks and thus provide both statistical and computational advantages over existing algorithms.

For the problem of learning graphs with a few hubs, we have proposed an estimator based on a *sufficiency measure* — a quantitative criteria that measures whether or not the given number of samples suffice to estimate the neighborhood of a given node. Since the number of samples depends on the high-degree nodes (among other things), this measure allows us to detect "high degree" or "hub" nodes. Ignoring the estimates for these nodes, and only using "non-hub" nodes to reconstruct the whole graph then suffices to estimate graphs without any "hub-hub" edges.

For the problem of kernel ridge regression, we have studied a divide-and-conquer approach to reduce the $O(n^3)$ computational complexity of solving a single KRR problem (where n is the number of samples). Our divide step is based on a *suitable* underlying partition of the input space (possibly obtained

via clustering), and the conquer step simply uses the local estimate of each partition as the overall estimate for that partition. We have studied conditions under which generalization rates can be obtained for such a partitioning based KRR estimator. Moreover, we have shown its statistical advantages over both a single KRR estimate and an estimator based on random data partitioning. This is explained by the fact that a partitioning based estimator learns a piecewise KRR estimates, thereby allowing the possibility of lower *approximation error* as well.

For the problem of stragglers/slow machines in distributed synchronous gradient descent, we have proposed *gradient coding* — a framework that replicates data blocks and codes across gradients. We have experimented with various *gradient coding* ideas on Amazon EC2 instances. Our proposed schemes create computation overheads while keeping communication the same. The benefit of this additional computation is fault-tolerance: we are able to recover full gradients, even if s machines do not deliver their assigned work, or are slow in doing so. Moreover, our partial straggler schemes provide fault tolerance while allowing all machines to do partial work. They however require an extra round of communication.

Appendices

Appendix A

Appendix A - Proofs for Chapter 2

A.1 Proof of Corollary 2.1

Proof. For any $t \in \mathcal{N}^*_{\text{sub}}(r)$, we have

$$\widehat{\mathcal{N}}_\lambda(r; D) = \mathcal{N}^*_{\text{sub}}(r) \Rightarrow t \in \widehat{\mathcal{N}}_\lambda(r; D). \tag{A.1}$$

For any $t \notin \mathcal{N}^*_{\text{sub}}(r)$, we have

$$t \in \widehat{\mathcal{N}}_\lambda(r; D) \Rightarrow \widehat{\mathcal{N}}_\lambda(r; D) \neq \mathcal{N}^*_{\text{sub}}(r). \tag{A.2}$$

Thus,

$$\begin{aligned} &\mathbb{P}(t \in \widehat{\mathcal{N}}_\lambda(r; D)) \geq \mathbb{P}(\widehat{\mathcal{N}}_\lambda(r; D) = \mathcal{N}^*_{\text{sub}}(r)) \quad \text{if } t \in \mathcal{N}^*_{\text{sub}}(r) \text{ and,} \\ &\mathbb{P}(t \in \widehat{\mathcal{N}}_\lambda(r; D)) \leq \mathbb{P}(\widehat{\mathcal{N}}_\lambda(r; D) \neq \mathcal{N}^*_{\text{sub}}(r)) \quad \text{if } t \notin \mathcal{N}^*_{\text{sub}}(r). \end{aligned} \tag{A.3}$$

Now, using the result of Theorem 2.1 proves the corollary. □

A.2 Proof of Proposition 2.1

Proof. The proof of this proposition is similar to Theorem 4.1 in [37]. First note that,

$$\mathbb{E}\left[\widetilde{p}_{r,b,\lambda}(t; D)\right] = \frac{1}{\binom{n}{b}} \sum_{D_b \in S_b(D)} \mathbb{E}\left[F^t_{\lambda,r}(D_b)\right] = \frac{1}{\binom{n}{b}} \sum_{D_b \in S_b(D)} \mathbb{P}\left(t \in \widehat{\mathcal{N}}_{b,\lambda}(r; D_b)\right), \tag{A.4}$$

where the expectation and probability are taken over the samples D being drawn i.i.d. For any fixed set of b indices, drawing n samples i.i.d. and then choosing the b samples corresponding to the fixed indices is equivalent to drawing b samples i.i.d. Thus, for any $D_b \in S_b(D)$, we have $\mathbb{P}\left(t \in \widehat{\mathcal{N}}_{b,\lambda}(r; D_b)\right) = p_{r,b,\lambda}(t)$, which implies

$$\mathbb{E}\left[\widetilde{p}_{r,b,\lambda}(t; D)\right] = p_{r,b,\lambda}(t). \tag{A.5}$$

Using Hoeffding's inequality for a U-statistics [53], we can concentrate $\widetilde{p}_{r,b,\lambda}(t; D)$ around its expectation as

$$\mathbb{P}\left(|\widetilde{p}_{r,b,\lambda}(t; D) - p_{r,b,\lambda}(t)| > \frac{\epsilon}{2}\right) \leq 2\exp\left(-\frac{n\epsilon^2}{2b}\right). \tag{A.6}$$

Now, consider $\widetilde{p}_{r,b,\lambda}(t; D)$ for a fixed set of samples D. We can think of $\widetilde{p}_{r,b,\lambda}(t; D)$ as the expected value of a random variable on a uniform distribution over subsets of size b *i.e.* imagine we have a random variable Y which can take values $F^t_{\lambda,r}(D_b)$ for $D_b \in S_b(D)$, and

$$\mathbb{P}\left(Y = F^t_{\lambda,r}(D_b)\right) = \frac{1}{\binom{n}{b}}, \tag{A.7}$$

so that $\widetilde{p}_{r,b,\lambda}(t; D) = \mathbb{E}[Y]$. Then, $\hat{p}_{r,b,\lambda}(t; D)$ is an estimate of $\mathbb{E}[Y]$, computed by averaging N values of Y, chosen independently and uniformly randomly. Using McDiarmid's inequality [40], we can therefore concentrate $\hat{p}_{r,b,\lambda}(t; D)$ around $\widetilde{p}_{r,b,\lambda}(t; D)$ as

$$\begin{aligned} &\mathbb{P}\left(|\hat{p}_{r,b,\lambda}(t; D) - \widetilde{p}_{r,b,\lambda}(t; D)| > \frac{\epsilon}{2} \,\middle|\, D\right) \leq 2\exp\left(-\frac{N\epsilon^2}{2}\right), \\ \Rightarrow\ &\mathbb{P}\left(|\hat{p}_{r,b,\lambda}(t; D) - \widetilde{p}_{r,b,\lambda}(t; D)| > \frac{\epsilon}{2}\right) \leq 2\exp\left(-\frac{N\epsilon^2}{2}\right), \end{aligned} \tag{A.8}$$

where we obtain the second inequality by integrating D out, since the RHS does not depend on D.

Combining Equation (A.6) and (A.8), we get

$$\mathbb{P}\Big(|\hat{p}_{r,b,\lambda}(t;D) - p_{r,b,\lambda}(t)| > \epsilon\Big) \leq 2\exp\left(-\frac{n\epsilon^2}{2b}\right) + 2\exp\left(-\frac{N\epsilon^2}{2}\right). \quad \text{(A.9)}$$

For, $N \geq \lceil \frac{n}{b} \rceil$, this becomes

$$\mathbb{P}\Big(|\hat{p}_{r,b,\lambda}(t;D) - p_{r,b,\lambda}(t)| > \epsilon\Big) \leq 4\exp\left(-\frac{n\epsilon^2}{2b}\right). \quad \text{(A.10)}$$

Now, by the union bound,

$$\begin{aligned}\mathbb{P}\Big(\exists\, t \in V \setminus r \text{ s.t. } |\hat{p}_{r,b,\lambda}(t;D) - p_{r,b,\lambda}(t)| > \epsilon\Big) &\leq 4(p-1)\exp\left(-\frac{n\epsilon^2}{2b}\right) \\ &\leq 4p\exp\left(-\frac{n\epsilon^2}{2b}\right)\end{aligned} \quad \text{(A.11)}$$

Finally, observe that $\exists t' \in V \setminus r$ s.t.

$$\begin{aligned}|\widehat{\mathcal{M}}_{r,b,\lambda}(D) - \mathcal{M}_{r,b,\lambda}| &= \Big|\max_{t_1 \in V\setminus r} \hat{p}_{r,b,\lambda}(t_1;D)\left(1-\hat{p}_{r,b,\lambda}(t_1;D)\right) - \max_{t_2 \in V\setminus r} p_{r,b,\lambda}(t_2)\left(1-p_{r,b,\lambda}(t_2)\right)\Big| \\ &\leq \Big|\hat{p}_{r,b,\lambda}(t';D)\left(1-\hat{p}_{r,b,\lambda}(t';D)\right) - p_{r,b,\lambda}(t')\left(1-p_{r,b,\lambda}(t')\right)\Big| \\ &\leq \Big|\hat{p}_{r,b,\lambda}(t';D) - p_{r,b,\lambda}(t')\Big| + \Big|\left(\hat{p}_{r,b,\lambda}(t';D) - p_{r,b,\lambda}(t')\right)\left(\hat{p}_{r,b,\lambda}(t';D) + p_{r,b,\lambda}(t')\right)\Big| \\ &\leq 3|\hat{p}_{r,b,\lambda}(t';D) - p_{r,b,\lambda}(t')|\end{aligned} \quad \text{(A.12)}$$

An instance of the t' used in the above set of inequations can be one of t_1^* or t_2^*, corresponding to the optimal for $\left(\arg\max_{t_1 \in V\setminus r} \hat{p}_{r,b,\lambda}(t_1;D)\left(1-\hat{p}_{r,b,\lambda}(t_1;D)\right)\right)$ and $\left(\arg\max_{t_2 \in V\setminus r} p_{r,b,\lambda}(t_2)\left(1-p_{r,b,\lambda}(t_2)\right)\right)$ respectively.

Thus,

$$|\widehat{\mathcal{M}}_{r,b,\lambda}(D) - \mathcal{M}_{r,b,\lambda}| > \epsilon \Rightarrow \exists t' \in V \setminus r \text{ s.t. } |\hat{p}_{r,b,\lambda}(t'; D) - p_{r,b,\lambda}(t')| > \epsilon/3 \quad \text{(A.13)}$$

Using the result of Equation (A.10) now proves the lemma. □

A.3 Proof of Proposition 2.2

Proof. Consider any $t \in V \setminus r$. From Assumption 2.1, we know that

$$\begin{aligned} &\forall\, \lambda \in [0, \lambda_{\min}(t))\,, \quad p_{r,b,\lambda}(t) > (1 - 2\exp(-c \log p)) \text{ and,} \\ &\forall\, \lambda \in [\lambda_{\min}(t), \lambda_{\max}(t)]\,, \quad 2\exp(-c \log p) \leq p_{r,b,\lambda}(t) \leq (1 - 2\exp(-c \log p))\,. \end{aligned} \quad \text{(A.14)}$$

This implies that

$$\begin{aligned} &\forall\, \lambda \in [0, \lambda_{\min}(t))\,, \quad p_{r,b,\lambda}(t)\,(1 - p_{r,b,\lambda}(t)) < \gamma \text{ and,} \\ &\forall\, \lambda \in [\lambda_{\min}(t), \lambda_{\max}(t)]\,, \quad p_{r,b,\lambda}(t)\,(1 - p_{r,b,\lambda}(t)) \geq \gamma. \end{aligned} \quad \text{(A.15)}$$

Suppose we pick $\lambda_l' = \min_{t \in V \setminus r} \lambda_{\min}(t)$. Then for all $\lambda < \lambda_l'$, $\mathcal{M}_{r,b,\lambda} < \gamma$, and at λ_l', $\mathcal{M}_{r,b,\lambda_l'} \geq \gamma$. This means that λ_l' is the solution to $\inf\{\lambda \geq 0 : \mathcal{M}_{r,b,\lambda} \geq \gamma\}$. Thus, $\lambda_l = \inf\{\lambda \geq 0 : \mathcal{M}_{r,b,\lambda} \geq \gamma\}$ exists and

$$\lambda_l = \lambda_l' = \min_{t \in V \setminus r} \lambda_{\min}(t). \quad \text{(A.16)}$$

To prove the existence of λ_u, we first have the following claim, the proof of which is described in Subsection A.3.1.

Claim A.1. *For any node $r \in V$, there exists a regularization parameter $\lambda_s\,(0 \leq \lambda_s \leq 1)$ s.t. for all $\lambda > \lambda_s$, $p_{r,b,\lambda}(t) = 0\,\forall\, t \in V \setminus r$, and as a consequence, $\mathcal{M}_{r,b,\lambda} = 0$.*

Now, observe that $\mathcal{M}_{r,b,\lambda}$ is a continuous function of λ, since $\mathcal{M}_{r,b,\lambda} = \max_{t \in V \setminus r} p_{r,b,\lambda}(t)\,(1 - p_{r,b,\lambda}(t))$ is just a maximum of continuous functions.

So, $\mathcal{M}_{r,b,\lambda_l} \geq \gamma$, $\mathcal{M}_{r,b,\lambda_s} = 0$ (from Claim A.1) and the continuity of $\mathcal{M}_{r,b,\lambda}$, together imply that $\lambda_u = \inf\{\lambda > \lambda_l : \mathcal{M}_{r,b,\lambda} < \gamma\}$ exists. Also, we have $\lambda_u \leq \lambda_s$.

Finally, (b) is a consequence of the continuity of $p_{r,b,\lambda}(t)$. From (A.16), we know that $\lambda_l = \min_{t \in V \setminus r} \lambda_{\min}(t)$. Therefore, at $t' = \arg\min_{t \in V \setminus r} \lambda_{\min}(t)$ we have

$$p_{r,b,\lambda_l}(t') = 1 - 2\exp\left(-c\log p\right). \tag{A.17}$$

Note that equality occurs due to continuity of $p_{r,b,\lambda}(t)$. At λ_u, since $\mathcal{M}_{r,b,\lambda_u} < \gamma$, we must have either $p_{r,b,\lambda_u}(t') > 1 - 2\exp(-c\log p)$ or $p_{r,b,\lambda}(t') < 2\exp(-c\log p)$. This means that either $\lambda_u < \lambda_{\min}(t')$ or $\lambda_u > \lambda_{\max}(t')$. However, since $\lambda_u > \lambda_l = \lambda_{\min}(t')$, we cannot have the former. Thus, $p_{r,b,\lambda_u}(t') < 2\exp(-c\log p)$.

So, to summarize,

$$\begin{aligned} &\text{At } \lambda_l,\ p_{r,b,\lambda_l}(t') = 1 - 2\exp\left(-c\log p\right) \text{ and} \\ &\text{at } \lambda_u,\ p_{r,b,\lambda_u}(t') < 2\exp(-c\log p), \end{aligned} \tag{A.18}$$

i.e. between λ_l and λ_u, $p_{r,b,\lambda}(t')$ goes from a value close to 1, to a value close to 0. Now, continuity of $p_{r,b,\lambda}(t')$ implies that for any $k \in (\gamma, 1/4]$, there exists a λ s.t. $p_{r,b,\lambda}(t')\,(1 - p_{r,b,\lambda}(t')) \geq k$, which implies $\mathcal{M}_{r,b,\lambda} \geq k$. □

A.3.1 Proof of Claim A.1

Proof. Let D be any set of b samples, $D = \{x^{(1)} \ldots, x^{(b)}\}$. Any solution, $\widetilde{\theta}_{\setminus r}$, of (7) (with the samples D) must satisfy

$$\nabla \mathcal{L}(\widetilde{\theta}_{\setminus r}; D) + \lambda z = 0 \tag{A.19}$$

for some $z \in \partial \|\widetilde{\theta}_{\setminus r}\|_1$.

Suppose we have $\lambda > \|\nabla \mathcal{L}(0; D)\|_\infty$ and we pick $z_i = -[\nabla \mathcal{L}(0; D)]_i / \lambda$. Then, $z \in \partial \|\widetilde{\theta}_{\setminus r}\|_1$ for $\widetilde{\theta}_{\setminus r} = 0$ and $(0, z)$ satisfies (A.19). Thus, 0 is an optimum for (7). Also, since we have shown the existence of a subgradient z s.t. $\|z\|_\infty < 1$, by Lemma 1 in [49] we know that 0 is the only solution. If we pick $\lambda_s = \max_{D \in \{-1,1\}^{pb}} \|\nabla \mathcal{L}(0; D)\|_\infty$, then for any $\lambda > \lambda_s$, 0 is the unique optimum for any choice of D. This implies that $p_{r,b,\lambda}(t) = 0 \, \forall t \in V \setminus r$ and $\mathcal{M}_{r,b,\lambda} = 0$. Finally, note that

$$\|\nabla \mathcal{L}(0; D)\|_\infty = \max_{t \in V \setminus r} \left| \frac{1}{n} \sum_{i=1}^{b} x_r^{(i)} x_t^{(i)} \right| \leq 1 \Rightarrow \lambda_s \leq 1 \tag{A.20}$$

□

A.4 Proof of Proposition 2.4

Proof. Consider any $t \in V \setminus r$. We have

$$\text{Either } \ \lambda_u < \lambda_{\min}(t) \ \text{ or } \ \lambda_u > \lambda_{\max}(t). \tag{A.21}$$

This can be seen as at λ_u, we have $\mathcal{M}_{r,b,\lambda_u} > \gamma = 2 \exp(-c \log p) \, (1 - 2 \exp(-c \log p))$. This implies that

$$\text{Either } \ p_{r,b,\lambda_u}(t) > 1 - 2 \exp(-c \log p) \ \text{ or } \ p_{r,b,\lambda_u}(t) < 2 \exp(-c \log p). \tag{A.22}$$

Based on Assumption 2.1(a), this implies equation (A.21).

Now, consider this for any two irrelevant variables $t_1, t_2 \notin \mathcal{N}^*(r)$. We cannot have $\lambda_u < \lambda_{\min}(t_1)$ and $\lambda_u > \lambda_{\max}(t_2)$ (or vice-versa), as this would violate Assumption 2.1(b). Thus, we must have

$$\text{Either } \lambda_u < \min_{t \notin \mathcal{N}^*(r)} \lambda_{\min}(t) \text{ or } \lambda_u > \max_{t \notin \mathcal{N}^*(r)} \lambda_{\max}(t). \tag{A.23}$$

We shall show that the former possibility cannot happen. To see this, assume $\lambda_u < \min_{t \notin \mathcal{N}^*(r)} \lambda_{\min}(t)$. Then, using Assumption 2.1(c), this means that $\lambda_u < \lambda_{\max}(\tilde{t})$, for any $\tilde{t} \in V \backslash r$. But, from (A.21), this must imply that $\lambda_u < \lambda_{\min}\left(\tilde{t}\right)$, for any $\tilde{t} \in V \backslash r$. However, this is a contradiction, since $\lambda_u > \lambda_l = \min_{t \in V \backslash r} \lambda_{\min}(t)$, where the equality comes through the same argument used to show (A.16).

Thus, $\lambda_u > \max_{t \notin \mathcal{N}^*(r)} \lambda_{\max}(t)$. This implies that $p_{r,b,\lambda_u}(t) < 2\exp(-c \log p)$ for any $t \notin \mathcal{N}^*(r)$ *i.e.*

$$\text{For any } t \notin \mathcal{N}^*(r),\ \mathbb{P}\left(t \notin \widehat{\mathcal{N}}_{b,\lambda_u}(r; D)\right) \geq 1 - 2\exp(-c \log p). \tag{A.24}$$

Using union bound on the irrelevant variables, we get that $\mathbb{P}\left(\widehat{\mathcal{N}}_{b,\lambda_u}(r; D) \subseteq \mathcal{N}^*(r)\right) \geq 1 - 2\exp\left(-(c-1)\log p\right)$. □

A.5 Proof of Proposition 2.3

Proof. Following the same argument as in Proposition 2.4 above, we can infer that for any $t \notin \mathcal{N}^*(r)$, $p_{r,b,\lambda_u}(t) < 2\exp(-c \log p)$.

Using Corollary 2.1, we know that there exists a λ_0 s.t.

$$p_{r,b,\lambda_0}(t) \geq 1 - 2\exp(-c_1 c_4 \log p) > 1 - 2\exp(-c \log p) \quad \text{if } t \in \mathcal{N}^*_{sub}(r)$$
$$p_{r,b,\lambda_0}(t) \leq 2\exp(-c_1 c_4 \log p) < 2\exp(-c \log p) \quad \text{if } t \notin \mathcal{N}^*_{sub}(r). \tag{A.25}$$

Based on Assumption 2.1, this means for any $t \in \mathcal{N}^*_{sub}(r)$ we have $\lambda_0 < \lambda_{\min}(t)$, and for any $t \notin \mathcal{N}^*_{sub}(r)$ we have $\lambda_0 > \lambda_{\max}(t)$.

Observe that $\lambda_0 > \lambda_l$. This is because for any $t' \notin \mathcal{N}^*_{sub}(r)$, $\lambda_0 > \lambda_{\max}(t')$ which implies $\lambda_0 > \lambda_{\min}(t')$, whereas $\lambda_l = \min_{t'' \in V \setminus r} \lambda_{\min}(t'')$, using arguments used to show (A.16).

Now, we shall show that we cannot have $\lambda_0 < \lambda_u$. Suppose $\lambda_0 < \lambda_u$. From (A.25), we have that $\mathcal{M}_{r,b,\lambda_0} < \gamma$, where γ is as defined in Assumption 1. So, we get $\lambda_0 \in (\lambda_l, \lambda_u)$ s.t. $\mathcal{M}_{r,b,\lambda_0} < \gamma$. This is a contradiction since $\lambda_u = \inf\{\lambda > \lambda_l : \mathcal{M}_{r,b,\lambda} < \gamma\}$. Therefore, we must have $\lambda_u \leq \lambda_0$.

So, for any $t \in \mathcal{N}^*_{sub}(r)$, $\lambda_u < \lambda_{\min}(t)$, which means that $p_{r,b,\lambda_u}(t) > 1 - 2\exp(-c \log p)$. Now, taking a union bound over the exclusion of all irrelevant variables and the inclusion of all variables in $\mathcal{N}^*_{sub}(r)$ proves the proposition.

□

A.6 Proof of Theorem 2.2

Since this is a simple corollary, we shall only provide an outline of the proof here. The conditions specified in the theorem ensure that Proposition 2.3 is true for any node $r \in V$ with degree, $d(r) \leq d$, and that, Proposition 2.4 is true for any other node. In addition, owing to the choice of n and

N, Proposition 2.2 guarantees that $\widehat{\mathcal{M}}_{r,b,\lambda}$ would be reliable estimate for $\mathcal{M}_{r,b,\lambda}$ upto a tolerance of ϵ w.h.p. Thus, running Algorithm 2.2, with the parameters specified, for all nodes would yield the $\mathcal{N}^*_{sub}(r)$ neighbourhoods of nodes with degree at most d, and yield subsets of the true neighbourhoods for the rest. E_d is defined to be the set of edges (u, v) such that atleast one of its endpoints is a node with degree at most d (say u), and the other belongs to the $\mathcal{N}^*_{sub}$ neighbourhood of the first (*i.e.* $v \in \mathcal{N}^*_{sub}(u)$). Then, if we consider the union of all neighbourhoods obtained from Algorithm 2.2, clearly, the set E_d gets recovered with high probability.

A.7 Proof of Corollary 2.2

This is again a simple consequence of Theorem 2.2. Under the conditions specified here, the set E_d, defined in Theorem 2.2, becomes the set of true edges E^*. Thus, we are guaranteed exact graph recovery in this setting.

Appendix B

Appendix B - Supplementary for Chapter 3

B.1 Main bounds and covariance control

In this section we state bounds on $\mathrm{Reg}_i(\lambda, \overline{\lambda})$, $\mathrm{Bias}_i(\lambda, n)$, $\mathbb{E}_D\left[\mathrm{Var}_i(\lambda, D)\right]$ from Lemma 3.1. However, first we require a supplemental result on the operator norm of the sample covariance error, under a suitable *whitening*. This is typical of such analysis, and is detailed in the subsection below. Thereafter, we present the bounds on the component terms.

B.1.1 Covariance control

Bounding the terms in Lemma 3.1 requires control of the operator norm of the error in the sample covariance, under a suitable *whitening*. Specifically, we need a bound on the quantity (for any $i \in [m]$ and some $k \geq 2$):

$$\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\right\|^k\right]^{1/k} := \mathrm{CovErr}_i(\lambda p_i, n, k) \tag{B.1}$$

where we use the shorthand: $\Sigma_{i,\lambda p_i} = (\Sigma_i + \lambda p_i I)$. A general bound on this can be found in Lemma C.1 in Appendix C.1.2. While the expression in Lemma C.1 is complicated, it can be specialized for specific kernels to obtain meaningful expressions, as also shown in the supplementary. We state these for a few cases below. Their proof is provided in Appendix C.1.2.1

Finite Rank Kernels. Suppose kernel K has finite rank r — examples include the linear and polynomial kernels. Then, for any $i \in [m]$ and $k > 2$, we get: $\text{CovErr}_i(\lambda p_i, n, k) = O\left(\frac{\sqrt{\log r}\, S_i(\lambda p_i)}{\sqrt{n}}\right) = O\left(\frac{r\sqrt{\log r}}{\sqrt{n}}\right)$

Kernels with exponential decay in eigenvalues. Suppose kernel K has exponentially decaying eigenvalues, $\lambda_j \leq c_1 \exp(-c_2 j^2)$ ($\forall j$, and constants $c_1, c_2 > 0$) — an example here is the Gaussian kernel. Then, for any $i \in [m]$, $k > 2$ and $\lambda p_i \geq \text{poly}(1/n)$, we get: $\text{CovErr}_i(\lambda p_i, n, k) = O\left(\frac{\sqrt{\log n\,(\log\log n)}}{\sqrt{n}}\right)$

Kernels with polynomial decay in eigenvalues. Suppose kernel K has polynomially decaying eigenvalues, $\lambda_j \leq c j^{-v}$ ($\forall j$, and constants $c > 0, v > 2$) — examples here include sobolev kernels with different orders. Then, for any $i \in [m]$, $k > 2$, and $\lambda p_i \geq \frac{1}{n^\alpha}$ for some constant $\alpha < \frac{v}{2} - 1$, we get: $\text{CovErr}_i(\lambda p_i, n, k) = O\left(\frac{\sqrt{\log n}}{n^{\frac{1}{2} - \frac{\alpha+1}{v}}}\right)$

Overall, it would be useful to think of $\text{CovErr}_i(\lambda p_i, n, k)$ to be scaling as $\widetilde{O}\left(n^{-1/2}\right)$. Consequently, in the bounds to follow, terms of the form $\text{CovErr}_i(\lambda p_i, n, k)^k$ scale as $\widetilde{O}(n^{-k/2})$ — and become negligible for sufficiently large k.

B.1.2 Main bounds

We can now provide bounds on the terms in Lemma 3.1. The following lemmas provide bounds on the Regularization error, Bias and Variance, for any partition $i \in [m]$, as given in Definition 2. We only state the lemma here using the $O(\cdot)$ notation. Precise statements can be found in Lemmas C.2, C.3 and C.4, in Appendices C.1.6, C.1.7 and C.1.8 respectively. Recall that

$p_i = \mathbb{P}(x \in C_i)$ and $f_{i,\overline{\lambda}}$ is the solution of Eq. 3.7. Additionally, we use the shorthand CE_i to denote $\text{CovErr}_i(\lambda p_i, n, k)$.

Lemma B.1 (Regularization, Bias and Variance)**.** *Consider any partition $i \in [m]$. Let $k \geq 2$ such that Assumption 3.1 holds for this k (with constant a_1), and Assumption 3.2 holds (with $A_i(\overline{\lambda}) \geq 0$). Also, suppose p_i satisfies (for any $i \in [m]$): $p_i = \Omega\left(\log n/n\right)$. Then,*

$$Reg_i(\lambda, \overline{\lambda}) = O\left(T_1\right) \tag{B.2}$$

$$Bias_i(\lambda, n) \leq O\left((CE_i)^2 \left(T_2 + T_3 + (CE_i)^k T_4 + (CE_i)^{k/2} T_5\right)\right) \tag{B.3}$$

$$\mathbb{E}_D\left[Var_i(\lambda, D)\right] \leq O\left(\frac{\sigma^2 S_i(\lambda p_i)}{n} + T_1 + T_2 + (CE_i)^k T_4 + (CE_i)^{k/2} T_5\right) \tag{B.4}$$

where we let: $T_1 = \frac{(\overline{\lambda}-\lambda)^2 p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2$, $\quad T_2 = \frac{\sqrt{p_i} S_i(\lambda p_i) A_i(\overline{\lambda})^2}{n}$, $\quad T_3 = \frac{T_1}{n} S_i(\lambda p_i)^2 + \frac{\lambda p_i \left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2}{n}$, $\quad T_4 = \left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{\lambda}$, *and* $T_5 = \frac{A_i(\overline{\lambda})^2}{\lambda\sqrt{p_i}}$

Note that Lemma B.1 has a minimum requirement on p_i, namely: $p_i = \Omega\left(\log n/n\right)$. However, this is minor since this essentially corresponds to each partition having $\Omega(\log n)$ samples. Also, this requirement can be potentially avoided under other restrictions for e.g. if the unknown function f^* is uniformly bounded *i.e.* $|f^*(x)| \leq M \,\forall\, x$. Now, to interpret the above bounds, recall from Appendix B.1.1 that $CE_i = \text{CovErr}_i(d, \lambda p_i, n)$ can scale as $\widetilde{O}\left(n^{-1/2}\right)$. Therefore, terms of the form $(CE_i)^k$ — which scale as $\widetilde{O}(n^{-k/2})$ — will be of lower order for a large enough k. Also, the bias bound has a $(CE_i)^2$ factor outside — an $\widetilde{O}(n^{-1})$ term. Indeed, in most cases, the

bias term (Eq (B.3)) turns out to be of a much lower order than the variance term (Eq (B.4)). Moreover, the first two terms in the variance bound (Eq (B.4)), and the bound for Reg_i (Eq (B.2)), become the overall dominating terms. Consequently, using Lemma 3.1, we have an overall scaling of: $\mathbb{E}_D\left[\text{Err}_i(\hat{f}_{i,\lambda})\right] \approx O\left(\text{Approx}_i(\overline{\lambda}) + \frac{(\overline{\lambda}-\lambda)^2 p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2 S_i(\lambda p_i)}{n}\right)$.

B.2 Additional Discussion of Assumption 3.1

Let us recall Assumption 3.1.

Assumption (Eigenfunction moments). Let $\{\lambda_j^i, v_j^i\}_{j=1}^{\infty}$ denote the eigenvalue-eigenfunction pairs for the covariance operator Σ_i. Then, $\forall i \in [m]$, $\forall j$ s.t. $\lambda_j^i \neq 0$, and for some constant $k \geq 2$, we assume $\mathbb{E}\left[\left(v_j^i(x)^2 \mathbb{1}\left(x \in C_i\right)/\lambda_j^i\right)^{2k}\right] \leq a_1^k$, where a_1 is a constant.

We note that we always have: $\mathbb{E}\left[v_j^i(x)^2 \mathbb{1}\left(x \in C_i\right)/\lambda_j^i\right] = \mathbb{E}\left[\left\langle v_j^i, \phi_x\right\rangle_{\mathcal{H}}^2 \mathbb{1}\left(x \in C_i\right)\lambda_j^i\right] = \left\langle v_j^i, \Sigma_i v_j^i\right\rangle \lambda_j^i = 1$ *i.e.* the first moment of $(v_j^i(x)^2 \mathbb{1}\left(x \in C_i\right)/\lambda_j^i)$ always exists. Thus, Assumption 3.1 simply enforces existence of higher moments of $(v_j^i(x)^2 \mathbb{1}\left(x \in C_i\right)/\lambda_j^i)$. This assumption may also be interpreted as requiring partition-wise sub-Gaussian behaviour (up to $2k$ moments) in the RKHS space, since its primary use is to bound the quantity $\mathbb{E}\left[\left\|(\Sigma_i + \lambda p_i I)^{-1/2}\phi_x\right\|_{\mathcal{H}}^{2k} \mathbb{1}\left(x \in C_i\right)\right]$. We detail this in the subsection below.

B.2.1 Control of $\mathbb{E}\left[\left\|(\Sigma_i + \lambda I)^{-1/2}\phi_x\right\|_{\mathcal{H}}^{2k} \mathbb{1}\left(x \in C_i\right)\right]$ via Assumption 3.1

In this section, we show how Assumption 3.1 guarantees a bound on $\mathbb{E}\left[\left\|(\Sigma_i + \lambda I)^{-1/2}\phi_x\right\|_{\mathcal{H}}^{2k} \mathbb{1}\left(x \in C_i\right)\right]$. Consider any $i \in [m]$. Let us assume that

Assumption 3.1 holds with parameters a_1 and $k(\geq 2)$.

Now, note that for any $x \in \mathcal{X}$, we have:

$$\begin{aligned}
\phi_x &= \sum_j v_j^i(x) v_j^i \\
\Rightarrow (\Sigma_i + \lambda I)^{-1/2}\phi_x &= \sum_j \frac{\sqrt{\lambda_j^i}}{\sqrt{\lambda_j^i + \lambda}} \frac{v_j^i(x)}{\sqrt{\lambda_j^i}} v_j^i \\
\Rightarrow \left\|(\Sigma_i + \lambda I)^{-1/2}\phi_x\right\|_{\mathcal{H}}^{2k} &= \left(\sum_j \frac{\lambda_j^i}{\lambda_j^i + \lambda} \frac{v_j^i(x)^2}{\lambda_j^i} v_j^i\right)^k \\
&= \left(\sum_k \lambda_k^i/(\lambda_k^i + \lambda)\right)^k \left(\sum_j \frac{\lambda_j^i/(\lambda_j^i + \lambda)}{\sum_k \lambda_k^i/(\lambda_k^i + \lambda)} \frac{v_j^i(x)^2}{\lambda_j^i}\right)^k \\
&= S_i(\lambda)^k \left(\sum_j \frac{\lambda_j^i/(\lambda_j^i + \lambda)}{S_i(\lambda)} \frac{v_j^i(x)^2}{\lambda_j^i}\right)^k \\
&\overset{(a)}{\leq} S_i(\lambda)^k \left(\sum_j \frac{\lambda_j^i/(\lambda_j^i + \lambda)}{S_i(\lambda)} \left(\frac{v_j^i(x)^2}{\lambda_j^i}\right)^k\right) \qquad \text{(B.5)}
\end{aligned}$$

where we have (a) using Jensen's inequality.

Thus, we have

$$\begin{aligned}
\mathbb{E}\left[\left\|(\Sigma_i + \lambda I)^{-1/2}\phi_x\right\|_{\mathcal{H}}^{2k} \mathbb{1}\left(x \in C_i\right)\right] &\leq S_i(\lambda)^k \mathbb{E}\left[\sum_j \frac{\lambda_j^i/(\lambda_j^i + \lambda)}{S_i(\lambda)} \left(\frac{v_j^i(x)^2}{\lambda_j^i}\right)^k\right] \\
&\leq S_i(\lambda)^k a_1^k \qquad \text{(B.6)}
\end{aligned}$$

where we have used Assumption 3.1 in the last step.

B.3 Generalization Error for Polynomial Kernels

Theorem B.1 (Kernels with polynomial eigenvalue decay)**.** *Let $f^* \in \mathcal{H}$ and suppose kernel K has polynomially decaying eigenvalues : $\lambda_j \leq cj^{-v}$ ($\forall j$, and*

constants $c > 0, v > 2$). Let m denote the number of partitions, and let $k \geq 2$ such that Assumption 3.1 holds for this k. Also, suppose $\lambda p_i \geq \frac{1}{n^\alpha}$ for some constant $0 < \alpha < \frac{v}{2} - 1$, and $\forall i \in [m]$. Then, the overall error for the DC-estimator $\hat{f}_C$ is given as:

$$\mathbb{E}_D\left[Err(\hat{f}_C)\right] = O\left(\lambda \left\|f^*\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{n} g(\lambda) S(\lambda) + m\left(\frac{\log n}{n^{1-\frac{2(\alpha+1)}{v}}}\right)^{k/2}\left(\left\|f^*\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{\lambda}\right)\right) \tag{B.7}$$

Now, if $m = O\left(\sqrt{\frac{n^{k-\frac{2k(\alpha+1)}{v}-\frac{4v}{v+1}}}{(\log n)^k}}\right)$ and Assumption 3.3 holds at $\lambda = 1/n^{\frac{v}{v+1}}$ and $p_i \geq \frac{1}{n^{\alpha-\frac{v}{v+1}}}$ ($\forall i \in [m]$), then the DC-estimator $\hat{f}_C$ achieves the optimal rate: $\mathbb{E}_D\left[Err(\hat{f}_C)\right] = O\left(\frac{1}{n^{\frac{v}{v+1}}}\right)$ at $\lambda = 1/n^{\frac{v}{v+1}}$.

Note that the requirement of $p_i \geq \frac{1}{n^{\alpha-\frac{v}{v+1}}}$ in the latter part of the above theorem implicitly entails: $\alpha > \frac{v}{v+1}$. This, when coupled with the requirement $\alpha < \frac{v}{2} - 1$ from the former part of the above theorem, can only be meaningful for $v > 1 + \sqrt{2} \approx 2.44$. Therefore, the latter part of Theorem B.1 is only applicable to slightly stronger polynomial decays than the former part (which holds for $v > 2$). Now, assuming $v > 1 + \sqrt{2}$, the additional requirement of $m = O\left(\sqrt{\frac{n^{k-\frac{2k(\alpha+1)}{v}-\frac{4v}{v+1}}}{(\log n)^k}}\right)$ is only meaningful for a sufficiently large k. In particular, for $k \geq \frac{4v^2}{(v+1)(v-2(\alpha+1))}$. When this happens, Theorem B.1 guarantees the optimal rate $\mathbb{E}_D\left[\mathrm{Err}(\hat{f}_C)\right] = O\left(\frac{1}{n^{\frac{v}{v+1}}}\right)$[67].

B.4 Quintuplet condition

The bound on the residual error $\mathcal{E}_C$ in Theorem 3.3 requires that $CE_i = \mathrm{CovErr}_i(\lambda p_i, n, k) = \widetilde{O}\left(n^{-1/2}\right)$ — which is indeed the case for the kernels dis-

cussed in Section B.1.1. Also, it requires that the quintuplet $(n, m, k, p_i, \lambda, \overline{\lambda})$, for any $i \in [m]$, satisfies:

$$m = O\left(\max\left(\lambda n^{\frac{k-2}{2}}, \frac{n^{\frac{k-2}{2}}}{\|f_{\overline{\lambda}}\|_{\mathcal{H}}^2}\right)\right), \quad p_i = \Omega\left(\min\left(\frac{m^2}{\lambda^2 n^{k-2}}, \frac{\mathrm{Approx}(f_{\overline{\lambda}})}{n^{k/2}\overline{\lambda}}\right)\right) \tag{B.8}$$

The above restrictions on m and p_i essentially guarantee that all terms involving CE_i^k in Lemma B.1 are of a lower order.

Appendix C

Appendix B - Proofs for Chapter 3

C.1 Proofs

This section contains the proofs of all theorems, lemmas and corollaries presented in this paper, as well as some figures and tables. First, we summarize some definitions and notations in the following subsection.

C.1.1 Definitions and Notation

We are given n samples $\mathbf{D} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, of the tuple (x, y) drawn i.i.d. from a distribution, $\overline{\mathcal{P}}$, on $\mathcal{X} \times \mathcal{Y}$. x (and x_i) is a random vector in the input space $\mathcal{X}$, also called the covariate. y (and y_i) is a random variable in the output space $\mathcal{Y}$, also called the response. The collection of sets $\{C_1, \ldots, C_m\}$ is used to denote a disjoint partition of the covariate space:

$$\mathcal{X} = \cup_{i=1}^{m} C_i \text{ and } C_i \cap C_j = \{\phi\}, \forall\, i, j \in [m] \tag{C.1}$$

Additionally, we restrict $\mathcal{Y} \subseteq \mathbb{R}$ and assume an additive noise model relating the response to the covariate *i.e.* for each $i \in [n]$:

$$y_i = f^*(x_i) + \eta_i. \tag{C.2}$$

where $f^* : \mathcal{X} \to \mathbb{R}$ is an *unknown* mapping of covariates in $\mathcal{X}$ to responses in $\mathbb{R}$, and η_i is the random noise corresponding to sample i. We assume that f^*

is square integreable with respect to the marginal of $\overline{\mathcal{P}}$ on $\mathcal{X}$. Equivalently, we can say f^* lies in the space $\mathcal{L}_2(\mathcal{X}, \mathcal{P}) = \{f : \mathcal{X} \to \mathbb{R} \mid \|f\|_{L_2}^2 = \mathbb{E}\left[f(x)^2\right] < \infty\}$, where $\mathcal{P}$ denotes the marginal of $\overline{\mathcal{P}}$ on the input space $\mathcal{X}$. The random noise is assumed to be zero mean with bounded variance *i.e.* $\mathbb{E}\left[\eta_i | x_i\right] = 0$ and $\mathbb{E}\left[\eta_i^2 | x_i\right] \leq \sigma^2$, $\forall\, i \in [n]$.

We are given a *continuous, symmetric, positive definite* kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. For any $x \in \mathcal{X}$, we define $\phi_x := K(x, \cdot)$. Then, the Reproducing Kernel Hilbert Space (RKHS) corresponding to kernel K is given as $\mathcal{H} = \overline{\text{span}}\{\phi_x,\, x \in \mathcal{X}\}$, with inner product defined as

$$\left\langle \sum_j \alpha_j \phi_{x_j}, \sum_k \beta_k \phi_{x_k} \right\rangle_{\mathcal{H}} = \sum_j \sum_k \alpha_j \beta_k K(x_j, x_k) \tag{C.3}$$

We require that the RKHS space $\mathcal{H} \subset L_2(\mathcal{X}, \mathcal{P})$ — which means $\forall x$, $\mathbb{E}_{y \sim \mathcal{P}}[K(x, y)^2] < \infty$ — a condition which is always true for several kernel classes, including Gaussian, Laplacian, or any trace class kernel w.r.t. $\mathcal{P}$.

The partition based empirical and population covariance operators are defined as (for partition C_i):

$$\hat{\Sigma}_i = \frac{1}{n} \sum_{j=1}^{n} (\phi_{x_j} \otimes \phi_{x_j}) \mathbb{1}\left(x_j \in C_i\right) \tag{C.4}$$

$$\Sigma_i = \mathbb{E}\left[(\phi_x \otimes \phi_x) \mathbb{1}\left(x \in C_i\right)\right], \tag{C.5}$$

where $\phi_x \otimes \phi_x$ denotes the operator $\phi_x \left\langle \phi_x, \cdot \right\rangle_{\mathcal{H}}$, and $\mathbb{1}\left(\cdot\right)$ denotes the indicator

function. Note that we have the relation:

$$\Sigma = \sum_{i=1}^{m} \Sigma_i \tag{C.6}$$

where $\Sigma = \mathbb{E}\left[\phi_x \otimes \phi_x\right]$, the overall covariance operator.

We let $\{\lambda_j^i, v_j^i\}_{j=1,\ldots,\infty}$ be the collection of eigenvalue-eigenfunction pairs for Σ_i. Then,

$$\Sigma_i = \sum_j \lambda_j^i (v_j^i \otimes v_j^i) \tag{C.7}$$

For any $d \in \mathbb{N}, d \geq 1$, we define P_d as the projection operator onto the first d eigenfunctions of Σ_i. Thus,

$$P_d = \sum_{j=1}^{d} v_j^i \otimes v_j^i \tag{C.8}$$

We denote by $\hat{\Sigma}_i^d$ and Σ_i^d, the projected low-rank empirical and population covariances (with rank $= d$), obtained using the operator P_d. Thus,

$$\hat{\Sigma}_i^d = \frac{1}{n} \sum_{j=1}^{n} (P_d \phi_{x_j} \otimes P_d \phi_{x_j}) \mathbb{1}\left(x_j \in C_i\right) \tag{C.9}$$

$$\Sigma_i^d = \sum_{j=1}^{d} \lambda_j^i (v_j^i \otimes v_j^i) \tag{C.10}$$

For any $\lambda > 0$, we define the following spectral sums:

$$S_i(\lambda) = \sum_{j=1}^{\infty} \frac{\lambda_j^i}{\lambda_j^i + \lambda},\ U_i(d, \lambda) = \sum_{j=1}^{d} \frac{\lambda_j^i}{\lambda_j^i + \lambda},\ L_i(d, \lambda) = \sum_{j>d} \frac{\lambda_j^i}{\lambda_j^i + \lambda} \tag{C.11}$$

Thus, $S_i(\lambda) = U_i(d,\lambda) + L_i(d,\lambda)$, for any $d \in \mathbb{N}$.

Finally, we also introduce the shorthand: $\Sigma_{i,\lambda} = (\Sigma_i + \lambda I)$, $\phi'_x = \Sigma_{i,\lambda}^{-1/2}\phi_x$ and $P_d^{\perp} = \sum_{j>d}(v_j^i \otimes v_j^i)$.

C.1.2 Moments of the operator norm for Covariance operators

In this section, we state a lemma providing a bound on the quantity $\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k}$, for some constant $k \geq 2$. Note that the norm here, $\|\cdot\|$, corresponds to the operator norm. This quantity appears repeatedly in other bounds, and therefore it is useful to have a lemma recording its bound, as stated below. The proof can be found in Section C.1.9. First, we introduce the following notion of truncated spectral sums for Σ_i. For any $d \geq 1$, we let:

$$L_i(d,\lambda) = \sum_{j=d+1}^{\infty} \frac{\lambda_j^i}{\lambda_j^i + \lambda} \tag{C.12}$$

$$U_i(d,\lambda) = \sum_{j=1}^{d} \frac{\lambda_j^i}{\lambda_j^i + \lambda} \tag{C.13}$$

Note that for any $d \geq 1$, we have: $L_i(d,\lambda) + U_i(d,\lambda) = S_i(\lambda)$, where $S_i(\lambda)$ is defined in Eq. (C.11).

Now, we have the following lemma providing the required bound.

Lemma C.1. *Consider any $d \in \mathbb{N}, d \geq 1$. Also, let $k \geq 2$ such that Assumption 3.1 holds for this k (with constant a_1). Then, we have*

$$\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k} \leq \mathit{CovErr}_i(d,\lambda,n,k) \tag{C.14}$$

where we have the following expression for $CovErr_i(d, \lambda, n, k)$*:*

$$CovErr_i(d,\lambda,n,k) = a_1 L_i(d,\lambda) + a_1\sqrt{L_i(d,\lambda)U_i(d,\lambda)} + \frac{a_1\sqrt{e\log d}\; U_i(d,\lambda)}{\sqrt{n}} + \frac{4e\log d\;\left(a_1 U_i(d,\lambda) + \frac{\lambda_1^i}{\lambda_1^i+\lambda}\right)}{n^{1-1/k}} + \frac{\lambda_{d+1}^i}{\lambda_{d+1}^i+\lambda} \tag{C.15}$$

Using the above lemma and applying Markov's inequality, we get the following simple corollary.

Corollary C.1. *Consider any* $d \in \mathbb{N}, d \geq 1$*, and let* $k \geq 2$ *such that Assumption 3.1 holds for this* k *(with constant* a_1*). Then, we have*

$$\mathbb{P}\left(\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\| \geq \frac{1}{2}\right)) \leq 2^k\,[CovErr_i(d,\lambda,n)]^k \tag{C.16}$$

C.1.2.1 Bounds on $\mathbf{CovErr}_i(d, \lambda p_i, n, k)$ for specific cases

While the expression in Eq. C.15 may seem complicated, it is possible to obtain concrete expressions for specific kernels through an appropriate choice of d, similar to the approach in [67]. The idea is to choose a d which makes the $L_i(d, \lambda p_i)$ terms negligible in Eq. C.15. We do this for a few cases below.

Finite Rank Kernels. Suppose kernel K has finite rank r — examples include the linear and polynomial kernels. Then, for any $i \in [m]$, the partition-wise covariance operator Σ_i is also finite rank. Thus, we can pick $d = r$ (in Eq. C.15), which gives $L_i(d, \lambda p_i) = 0$ and $\lambda_{d+1}^i = 0$. Also, $U_i(d, \lambda p_i) = S_i(\lambda p_i) \leq r$.

Plugging these into Eq. C.15, we get:

$$\text{CovErr}_i(r, \lambda p_i, n) = O\left(\frac{\sqrt{\log r}\, S_i(\lambda p_i)}{\sqrt{n}}\right) = O\left(\frac{r\sqrt{\log r}}{\sqrt{n}}\right) \tag{C.17}$$

Kernels with polynomial decay in eigenvalues. Suppose kernel K has polynomially decaying eigenvalues, $\lambda_j \leq cj^{-v}$ ($\forall j$, and constants $c > 0, v > 2$) — examples here include sobolev kernels with different orders. Now, since we have $\Sigma = \sum_{i=1}^{m} \Sigma_i$ being a sum of psd operators, the minimax characterization of eigenvalues yields: $\lambda_j^i \leq \lambda_j \ \forall j$ and any $i \in [m]$. As a consequence, we have: $L_i(d, \lambda) \leq \sum_{j>d} \frac{\lambda_j}{\lambda_j + \lambda}$ and $S_i(\lambda) \leq S(\lambda)$. Then, following the same approach as [67] i.e. choosing $d = n^{C/(v-1)}$ for some constant $C > 0$, we get:

$$L_i(d, \lambda p_i) \leq \int_d^\infty \frac{cj^{-v}}{cj^{-v} + \lambda p_i} dj \leq \frac{c}{\lambda p_i} \int_d^\infty j^{-v} dj \leq \frac{c(v-1)}{\lambda p_i} d^{-(v-1)} \leq \frac{c(v-1)}{\lambda p_i n^C} \tag{C.18}$$

and, $U_i(d, \lambda p_i) \leq d = n^{C/(v-1)}$. Consequently, for $v > 2$ and $\lambda p_i \geq \frac{1}{n^{C\frac{v}{v-1}-1}}$, we get:

$$\text{CovErr}_i(n^{C/(v-1)}, \lambda p_i, n) = O\left(\frac{\sqrt{\log n}}{n^{\frac{1}{2} - \frac{C}{v-1}}}\right) \tag{C.19}$$

Kernels with exponential decay in eigenvalues. Suppose kernel K has exponentially decaying eigenvalues, $\lambda_j \leq c_1 \exp(-c_2 j^2)$ ($\forall j$, and constants $c_1, c_2 > 0$) — an example here is the Gaussian kernel. Again, since $\Sigma = \sum_{i=1}^{m} \Sigma_i$, the minimax characterization of eigenvalues yields: $\lambda_j^i \leq \lambda_j \ \forall j$ and any $i \in [m]$. Thus: $L_i(d, \lambda) \leq \sum_{j>d} \frac{\lambda_j}{\lambda_j + \lambda}$ and $S_i(\lambda) \leq S(\lambda)$. Choosing

$d = C\sqrt{\log n}/\sqrt{c_2}$ for some constant C, we get:

$$L_i(d, \lambda p_i) \leq \int_d^\infty \frac{c_1 \exp(-c_2 j^2)}{c_1 \exp(-c_2 j^2) + \lambda p_i} dj \leq \frac{c_1}{\lambda p_i} \int_d^\infty \exp(-c_2 j^2) dj$$
$$\leq \frac{c_1}{\lambda p_i} \exp(-c_2 d^2) \leq \frac{c_1}{\lambda p_i n^C} \quad \text{(C.20)}$$

and, $U_i(d, \lambda p_i) \leq d = C\sqrt{\log n}/\sqrt{c_2}$. Consequently, as long as $\lambda p_i \geq \text{poly}(1/n)$, we can choose a sufficiently large C to make the terms involving λ_{d+1}^i and $L_i(d, \lambda p_i)$ negligible. Thus, we get:

$$\text{CovErr}_i(C\sqrt{\log n}, \lambda p_i, n) = O\left(\frac{\sqrt{\log n \, (\log \log n)}}{\sqrt{n}}\right) \quad \text{(C.21)}$$

C.1.3 Proof of Lemma 3.1

The proof is as follows:

$$\begin{aligned}
\text{Err}_i(\hat{f}_{i,\lambda}) &= \mathbb{E}\left[(f^*(x) - \hat{f}_{i,\lambda}(x))^2\right] \\
&= \mathbb{E}\left[\left(f^*(x) - f_{i,\overline{\lambda}}(x) + f_{i,\overline{\lambda}}(x) - f_{i,\lambda}(x) + f_{i,\lambda}(x) - \hat{f}_{i,\lambda}(x)\right)^2\right] \\
&\overset{(a)}{\leq} 2\left(\text{Approx}_i(\overline{\lambda}) + 2\text{Reg}_i(\overline{\lambda}, \lambda) + 2\mathbb{E}\left[\left(f_{i,\lambda}(x) - \hat{f}_{i,\lambda}(x)\right)^2\right]\right)
\end{aligned} \quad \text{(C.22)}$$

where we have (a) since $(a+b+c)^2 \leq 2(a^2 + 2b^2 + 2c^2)$.

Now, following a standard bias-variance decomposition, we have:

$$\begin{aligned}
\mathbb{E}_D\left[\mathbb{E}\left[\left(f_{i,\lambda}(x) - \hat{f}_{i,\lambda}(x)\right)^2\right]\right] &= \mathbb{E}\left[\left(f_{i,\lambda}(x) - \bar{f}_{i,\lambda}(x)\right)^2\right] + \mathbb{E}_D\left[\mathbb{E}\left[\left(f_{i,\lambda}(x) - \hat{f}_{i,\lambda}(x)\right)^2\right]\right] \\
&= \text{Bias}_i(\lambda, n) + \mathbb{E}_D\left[\text{Var}_i(\lambda, D)\right]
\end{aligned} \quad \text{(C.23)}$$

Combining the above expressions, we get:

$$\mathbb{E}_D\left[\mathrm{Err}_i(\hat{f}_{i,\lambda})\right] \leq 2\left[\mathrm{Approx}_i(\overline{\lambda}) + 2\mathrm{Reg}_i(\overline{\lambda},\lambda) + 2\mathrm{Bias}_i(\lambda,n) + 2\mathbb{E}_D\left[\mathrm{Var}_i(\lambda,D)\right]\right] \tag{C.24}$$

C.1.4 Proof of Theorems 3.1, 3.2 and B.1

The theorems are a simple consequence of combining Lemmas C.2, C.3, and C.4 via Lemma 3.1, plugging $A_i(\overline{\lambda}) = 0$ with $\overline{\lambda} = 0$, ignoring the bias terms which are of a lower order, and using the expressions for $CE_i = \mathrm{CovErr}_i(\lambda p_i, n, k)$ discussed in Appendix B.1.1.

C.1.5 Proof of Theorem 3.3

Consider any $\overline{\lambda} > 0$, and let $f_{\overline{\lambda}}$ be the solution of Eq. (3.19). Now, for any partition $i \in [m]$, consider the following optimization problem:

$$\hat{f}_i = \underset{f\in\mathcal{H}, \|f\|_{\mathcal{H}} \leq \left\|f_{\overline{\lambda}}\right\|_{\mathcal{H}}}{\arg\min} \mathbb{E}\left[(f^*(x) - f(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \tag{C.25}$$

By duality, $\exists\ \overline{\lambda}_i' \geq 0$ s.t. $\hat{f}_i = f_{i,\overline{\lambda}_i'}$, with $f_{i,\overline{\lambda}_i'}$ being the solution of Eq. (3.7). Now, by the optimality of $f_{i,\overline{\lambda}_i'}$, we have:

$$\begin{aligned}\mathrm{Approx}_i(\overline{\lambda}_i') &= \mathbb{E}\left[(f^*(x) - f_{i,\overline{\lambda}_i'}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \\ &\leq \mathbb{E}\left[(f^*(x) - f_{\overline{\lambda}}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] = \mathrm{ApproxError}_i(f_{\overline{\lambda}})\end{aligned} \tag{C.26}$$

and $\left\|f_{i,\overline{\lambda}_i'}\right\|_{\mathcal{H}} \leq \|f_{\overline{\lambda}}\|_{\mathcal{H}}$.

Now, if $\overline{\lambda}_i' \leq \overline{\lambda}$, we are done. Suppose $\overline{\lambda}_i' > \overline{\lambda}$. Then, we know that $\mathrm{Approx}_i(\overline{\lambda}) \leq \mathrm{ApproxError}(f_{\overline{\lambda}})$, since decreasing the regularization penalty

from $\overline{\lambda}_i'$ to $\overline{\lambda}$ would only decrease the approximation error. Moreover, using the fact that the following function

$$T(\lambda) := \min_{f \in \mathcal{H}} \mathbb{E}\left[(f^*(x) - f(x))^2 \mathbb{1}\left(x \in C_i\right)\right] + \lambda p_i \left\|f\right\|_{\mathcal{H}}^2 \tag{C.27}$$

is a monotonically increasing function of λ[55], we have:

$$\begin{aligned} \text{Approx}_i(\overline{\lambda}) + \lambda p_i \left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 &\leq \text{Approx}_i(\overline{\lambda}_i') + \lambda p_i \left\|f_{i,\overline{\lambda}_i'}\right\|_{\mathcal{H}}^2 && \text{(C.28)} \\ &\leq \text{ApproxError}_i(f_{\overline{\lambda}}) + \lambda p_i \left\|f_{\overline{\lambda}}\right\|_{\mathcal{H}}^2 && \text{(C.29)} \end{aligned}$$

Thus, $\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}} = O\left(\|f_{\overline{\lambda}}\|_{\mathcal{H}} + \sqrt{\frac{\text{ApproxError}_i(f_{\overline{\lambda}})}{\overline{\lambda} p_i}}\right)$. Therefore, the result holds with $\overline{\lambda}_i = min(\overline{\lambda}, \overline{\lambda}_i')$.

The bound on the estimation error $\mathcal{E}_C$ is a simple consequence of the fact that, under the conditions assumed, all terms in Lemma B.1 involving $(CE)_i^k$ are of a lower order, and that the condition $\text{Approx}(\overline{\lambda}) = O(\overline{\lambda} \left\|f_{\overline{\lambda}}\right\|_{\mathcal{H}}^2)$ guarantees that:

$$\sum_i p_i \left\|f_{i,\overline{\lambda}_i}\right\|_{\mathcal{H}}^2 = O(\|f_{\overline{\lambda}}\|_{\mathcal{H}}^2) \tag{C.30}$$

Then, combining Lemma B.1 via Lemma 3.1 gives us the required scaling.

C.1.6 Regularization Bound

In this section we provide a proof of the bound on $\text{Reg}_i(\lambda, \overline{\lambda})$ in Lemma B.1. The bound with exact constants is stated below.

Lemma C.2. *For any $\lambda > 0, \overline{\lambda} > 0$ and partition $i \in [m]$,*

$$Reg_i(\lambda, \overline{\lambda}) = \mathbb{E}\left[(f_{i,\overline{\lambda}}(x) - f_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \leq p_i \frac{(\overline{\lambda} - \lambda)^2}{\lambda} \left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 \tag{C.31}$$

C.1.6.1 Proof of Lemma C.2

Proof. We want to bound

$$\begin{aligned}\mathbb{E}\left[(f_{i,\overline{\lambda}}(x) - f_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] &= \left\|f_{i,\overline{\lambda}} - f_{i,\lambda}\right\|_{\Sigma_i}^2 \\ &= \left\|\Sigma_i^{1/2}(f_{i,\overline{\lambda}} - f_{i,\lambda})\right\|_{\mathcal{H}}^2 \end{aligned} \tag{C.32}$$

Using first order conditions for the optimality of $f_{i,\lambda}$ and $f_{i,\overline{\lambda}}$, we have

$$\begin{aligned}(\Sigma_i + \lambda p_i I) f_{i,\lambda} &= \mathbb{E}\left[y\phi_x \mathbb{1}\left(x \in C_i\right)\right] \\ (\Sigma_i + \overline{\lambda} p_i I) f_{i,\overline{\lambda}} &= \mathbb{E}\left[y\phi_x \mathbb{1}\left(x \in C_i\right)\right] \end{aligned} \tag{C.33}$$

Thus, $f_{i,\lambda} = (\Sigma_i + \lambda p_i I)^{-1}(\Sigma_i + \overline{\lambda} p_i I) f_{i,\overline{\lambda}}$.

Letting $f_{i,\overline{\lambda}} = \sum_j \alpha_j v_j^i$, we get

$$\begin{aligned}\Sigma_i^{1/2}(f_{i,\overline{\lambda}} - f_{i,\lambda}) &= p_i(\lambda - \overline{\lambda}) \sum_j \frac{\sqrt{\lambda_j^i}}{\lambda_j^i + \lambda p_i} \alpha_j v_j^i \\ \Rightarrow \left\|\Sigma_i^{1/2}(f_{i,\overline{\lambda}} - f_{i,\lambda})\right\|_{\mathcal{H}}^2 &= p_i^2(\lambda - \overline{\lambda})^2 \sum_j \frac{\lambda_j^i}{(\lambda_j^i + \lambda p_i)^2} \alpha_j^2 \\ &\leq p_i \frac{(\lambda - \overline{\lambda})^2}{\lambda} \sum_j \alpha_j^2 = p_i \frac{(\lambda - \overline{\lambda})^2}{\lambda} \left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 \end{aligned} \tag{C.34}$$

□

C.1.7 Bias Bound

In this section we provide a proof of the bound on $\text{Bias}_i(\lambda, n)$ in Lemma B.1. The bound with exact constants is stated below.

Lemma C.3. *Consider any $d \in \mathbb{N}, d \geq 1$, and $k \geq 2$. Suppose Assumption 3.1 holds for this k (with constant a_1), and Assumption 3.2 holds. Also, suppose $\forall i \in [m]$, p_i satisfies: $p_i \geq \frac{16 \log(np_i)}{n-1}$. Then we have*

$$\begin{aligned} Bias_i(\lambda, n) \leq & (CovErr_i(d, \lambda p_i, n))^2 \times \\ & \left(T_1 + T_2 + 2^{k+1} \left[CovErr_i(d, \lambda p_i, n) \right]^k T_3 + 2^{k/2+3} \left[CovErr_i(d, \lambda p_i, n) \right]^{k/2} T_4 \right) \end{aligned} \tag{C.35}$$

where we let

$$\begin{aligned} T_1 &= \frac{16 a_1 \sqrt{p_i} S_i(\lambda p_i) A_i(\overline{\lambda})^2}{n} \\ T_2 &= \left(\frac{16 a_1^2 (\lambda - \overline{\lambda})^2}{\lambda} \frac{p_i S_i(\lambda p_i)^2 \left\| f_{i,\overline{\lambda}} \right\|_{\mathcal{H}}^2}{n} + \frac{8 \lambda p_i \left\| f_{i,\overline{\lambda}} \right\|_{\mathcal{H}}^2}{n} \right) \\ T_3 &= \left(2 \left\| f_{i,\overline{\lambda}} \right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{\lambda} \right) (\lambda_1^i + \lambda p_i) \\ T_4 &= \frac{(\lambda_1^i + \lambda p_i) A_i(\overline{\lambda})^2}{\lambda \sqrt{p_i}} \end{aligned} \tag{C.36}$$

C.1.7.1 Proof of Lemma C.3

Proof. We want to bound $\text{Bias}_i(\lambda, n)$, where

$$\begin{aligned} \text{Bias}_i(\lambda, n) &= \mathbb{E}\left[(f_{i,\lambda}(x) - \bar{f}_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right) \right] \\ &= \mathbb{E}\left[\left(\left\langle f_{i,\lambda} - \bar{f}_{i,\lambda}, \phi_x \right\rangle_{\mathcal{H}} \right)^2 \mathbb{1}\left(x \in C_i\right) \right] \\ &= \left\| f_{i,\lambda} - \bar{f}_{i,\lambda} \right\|_{\Sigma_i}^2 \end{aligned} \tag{C.37}$$

Let $\Delta_b = f_{i,\lambda} - \hat{f}_{i,\lambda}$. Then, equivalently, we want to bound $\left\| \mathbb{E}\left[\Delta_b\right] \right\|_{\Sigma_i}^2$

Now, from first order conditions of optimality for Eqs. (3.2) and (3.8),

we have

$$(\hat{\Sigma}_i + \lambda p_i I)\hat{f}_{i,\lambda} = \widehat{\mathbb{E}}\left[y\phi_x \mathbb{1}\left(x \in C_i\right)\right]$$
$$(\Sigma_i + \lambda p_i I) f_{i,\lambda} = \mathbb{E}\left[y\phi_x \mathbb{1}\left(x \in C_i\right)\right] \tag{C.38}$$

Combining the above, we get

$$\begin{aligned}\mathbb{E}\left[(\hat{\Sigma}_i + \lambda p_i I)\Delta_b\right] &= \mathbb{E}\left[(\hat{\Sigma}_i + \lambda p_i I) f_{i,\lambda}\right] - \mathbb{E}\left[(\hat{\Sigma}_i + \lambda p_i I)\hat{f}_{i,\lambda}\right] \\ &= (\Sigma_i + \lambda p_i I) f_{i,\lambda} - \mathbb{E}\left[y\phi_x \mathbb{1}\left(x \in C_i\right)\right] \\ &= 0 \end{aligned} \tag{C.39}$$

Rearranging and multiplying $\Sigma_{i,\lambda p_i}^{-1/2}$, we get

$$\begin{aligned}\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b\right] &= -\mathbb{E}\left[\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\Sigma_{i,\lambda p_i}^{1/2}\Delta_b\right] \\ &= -\mathbb{E}\left[\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right] \end{aligned} \tag{C.40}$$

where we let X denote the set $\{x_1, \ldots, x_n\}$ *i.e.* the covariates in the data $\mathbf{D}$.

So,

$$\begin{aligned}\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b\right]\right\|_{\mathcal{H}}^2 &= \left\|\mathbb{E}\left[\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right]\right\|_{\mathcal{H}}^2 \\ \Rightarrow \left\|\Sigma_i^{1/2}\mathbb{E}\left[\Delta_b\right]\right\|_{\mathcal{H}}^2 \overset{(a)}{\leq} \left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b\right]\right\|_{\mathcal{H}}^2 &= \left\|\mathbb{E}\left[\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right]\right\|_{\mathcal{H}}^2 \\ &\overset{(b)}{\leq} \left(\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right\|_{\mathcal{H}}\right]\right)^2 \\ &\overset{(c)}{\leq} \left(\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\right\|_{\mathcal{H}}\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right\|_{\mathcal{H}}\right]\right)^2 \\ &\overset{(d)}{\leq} \mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\right\|_{\mathcal{H}}^2\right]\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right\|_{\mathcal{H}}^2\right] \end{aligned} \tag{C.41}$$

where we have (a) using the fact that $\langle u, \Sigma_i u\rangle_{\mathcal{H}} < \langle u, (\Sigma_i + \lambda p_i I)u\rangle_{\mathcal{H}}\ \forall u \in \mathcal{H}$, (b) by Jensen's inequality, (c) by the definition of the operator norm, (d) by the Cauchy-Schwarz inequality.

Thus,

$$\|\mathbb{E}\left[\Delta_b\right]\|_{\Sigma_i}^2 \leq \mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\right\|_{\mathcal{H}}^2\right] \mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right\|_{\mathcal{H}}^2\right] \tag{C.42}$$

Now, Lemma C.1 provides a bound for $\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\right\|_{\mathcal{H}}^2\right]$. For the remainder of the proof, we provide the bound for $\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right\|_{\mathcal{H}}^2\right]$. Combining these bounds will yield the main statement of the lemma.

From first order conditions again (Eq. (C.38)), we have

$$(\hat{\Sigma}_i + \lambda p_i I)\mathbb{E}\left[\Delta_b \,|\, X\right] = \widehat{\mathbb{E}}\left[f^*(x)\phi_x \mathbb{1}\left(x \in C_i\right)\right] - (\hat{\Sigma}_i + \lambda p_i) f_{i,\lambda} \tag{C.43}$$

Multiplying by $\Sigma_{i,\lambda p_i}^{-1/2}$ on both sides and rewriting differently, we get

$$\begin{aligned}
&\left(\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2} + I\right)\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right] \\
&= \Sigma_{i,\lambda p_i}^{-1/2}\left(\widehat{\mathbb{E}}\left[f^*(x)\phi_x \mathbb{1}\left(x \in C_i\right)\right] - (\hat{\Sigma}_i + \lambda p_i) f_{i,\lambda}\right) \\
&= \left(\widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\lambda}(x))\Sigma_{i,\lambda p_i}^{-1/2}\phi_x \mathbb{1}\left(x \in C_i\right)\right] - \lambda p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\lambda}\right) \qquad \text{(C.44)} \\
\Rightarrow \quad &\left\|\left(\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2} + I\right)\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \,|\, X\right]\right\|_{\mathcal{H}}^2 \\
&= \left\|\widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\lambda}(x))\Sigma_{i,\lambda p_i}^{-1/2}\phi_x \mathbb{1}\left(x \in C_i\right)\right] - \lambda p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\lambda}\right\|_{\mathcal{H}}^2 \\
&= \left\|\frac{1}{n}\sum_{j=1}^{n} w_j\right\|_{\mathcal{H}}^2 \qquad \text{(C.45)}
\end{aligned}$$

where we define $w_j := (f^*(x_j) - f_{i,\lambda}(x_j))\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\mathbb{1}\left(x_j \in C_i\right) - \lambda p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\lambda}$. Note that $\mathbb{E}\left[w_j\right] = 0$.

Let us define the event $\mathcal{E}_{cov} = \left\{\left\|\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2}\right\| \leq 1/2\right\}$. Note that from Corollary C.1, we have $\mathbb{P}\left(\mathcal{E}_{cov}^c\right) \leq 2^k \left[\mathrm{CovErr}_i(d, \lambda p_i, n)\right]^k$. Now, under the event $\mathcal{E}_{cov}$,

$$\begin{aligned}\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \mid X\right]\right\|_{\mathcal{H}}^2\right] &\leq 4\mathbb{E}\left[\left\|\frac{1}{n}\sum_{j=1}^n w_j\right\|_{\mathcal{H}}^2\right] \\ &= \frac{4}{n^2}\sum_{j=1}^n \mathbb{E}\left[\|w_j\|_{\mathcal{H}}^2\right] \end{aligned} \tag{C.46}$$

To control $\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \mid X\right]\right\|_{\mathcal{H}}^2\right]$ overall, we have

$$\begin{aligned}\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \mid X\right]\right\|_{\mathcal{H}}^2\right] &= \mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \mid X\right]\right\|_{\mathcal{H}}^2 \mathbb{1}\left(\mathcal{E}_{cov}\right)\right] + \mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\mathbb{E}\left[\Delta_b \mid X\right]\right\|_{\mathcal{H}}^2 \mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right] \\ &\leq \frac{4}{n^2}\sum_{j=1}^n \mathbb{E}\left[\|w_j\|_{\mathcal{H}}^2\right] + (\lambda_1^i + \lambda p_i)\mathbb{E}\left[\mathbb{E}\left[\|\Delta_b\|_{\mathcal{H}}^2 \mid X\right]\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right] \end{aligned} \tag{C.47}$$

Bound on $\mathbb{E}\left[\|w_j\|_{\mathcal{H}}^2\right]$. We have

$$\begin{aligned}
\mathbb{E}\left[\|w_j\|_{\mathcal{H}}^2\right] &\overset{(a)}{\leq} 2\mathbb{E}\left[(f^*(x_j)-f_{i,\lambda}(x_j))^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^2\mathbb{1}\left(x_j\in C_i\right)\right]+2(\lambda p_i)^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}f_{i,\lambda}\right\|_{\mathcal{H}}^2\\
&\overset{(b)}{\leq} 4\mathbb{E}\left[(f^*(x_j)-f_{i,\overline{\lambda}}(x_j))^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^2\mathbb{1}\left(x_j\in C_i\right)\right]\\
&\quad+4\mathbb{E}\left[(f_{i,\overline{\lambda}}(x_j)-f_{i,\lambda}(x_j))^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^2\mathbb{1}\left(x_j\in C_i\right)\right]+2(\lambda p_i)^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}f_{i,\lambda}\right\|_{\mathcal{H}}^2\\
&\overset{(c)}{\leq} 4\sqrt{\mathbb{E}\left[(f^*(x_j)-f_{i,\overline{\lambda}}(x_j))^4\mathbb{1}\left(x_j\in C_i\right)\right]}\sqrt{\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^4\mathbb{1}\left(x_j\in C_i\right)\right)\right]}\\
&\quad+4\left\|f_{i,\overline{\lambda}}-f_{i,\lambda}\right\|_{\Sigma_{i,\lambda p_i}}^2\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^4\mathbb{1}\left(x_j\in C_i\right)\right)\right]+2(\lambda p_i)^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}f_{i,\lambda}\right\|_{\mathcal{H}}^2\\
&\overset{(d)}{\leq} 4a_1\sqrt{p_i}A_i(\overline{\lambda})^2S_i(\lambda p_i)+4a_1^2S_i(\lambda p_i)^2\left\|f_{i,\overline{\lambda}}-f_{i,\lambda}\right\|_{\Sigma_{i,\lambda p_i}}^2+2(\lambda p_i)^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}f_{i,\lambda}\right\|_{\mathcal{H}}^2\\
&\overset{(e)}{\leq} 4a_1\sqrt{p_i}A_i(\overline{\lambda})^2S_i(\lambda p_i)+4a_1^2p_i\frac{(\lambda-\overline{\lambda})^2}{\lambda}S_i(\lambda p_i)^2\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2+2\lambda p_i\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2\\
&=\left[4a_1\sqrt{p_i}A_i(\overline{\lambda})^2\right]S_i(\lambda p_i)+\left[4a_1^2p_i\frac{(\lambda-\overline{\lambda})^2}{\lambda}S_i(\lambda p_i)^2+2\lambda p_i\right]\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2
\end{aligned}\tag{C.48}$$

where we have (a) using $\|x+y\|_{\mathcal{H}}^2\leq 2\|x\|_{\mathcal{H}}^2+2\|y\|_{\mathcal{H}}^2$, (b) since $(f^*(x_j)-f_{i,\lambda}(x_j))^2\leq 2(f^*(x_j)-f_{i,\overline{\lambda}}(x_j))^2+2(f_{i,\overline{\lambda}}(x_j)-f_{i,\lambda}(x_j))^2$, (c) using Cauchy-Schwarz inequality in two different ways, namely, $\mathbb{E}\left[XY\right]\leq\sqrt{\mathbb{E}\left[X^2\right]}\sqrt{\mathbb{E}\left[Y^2\right]}$ and $(f_{i,\overline{\lambda}}(x_j)-f_{i,\lambda}(x_j))^2=\left(\left\langle f_{i,\overline{\lambda}}-f_{i,\lambda},\phi_{x_j}\right\rangle_{\mathcal{H}}\right)^2\leq\left\|f_{i,\overline{\lambda}}-f_{i,\lambda}\right\|_{\Sigma_{i,\lambda p_i}}^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^2$, (d) using Assumption 3.2, and via Jensen's inequality and Assumption 3.1

$$\begin{aligned}
\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^4\mathbb{1}\left(x_j\in C_i\right)\right)\right]&=\mathbb{E}\left[\left(\sum_j\frac{\lambda_j^i}{\lambda_j^i+\lambda p_i}\frac{v_j^i(x)^2}{\lambda_j^i}\mathbb{1}\left(x_j\in C_i\right)\right)^2\right]\\
&\leq S_i(\lambda p_i)^2\sum_j\frac{\lambda_j^i/(\lambda_j^i+\lambda p_i)}{\sum_k\lambda_k^i/(\lambda_k^i+\lambda p_i)}\mathbb{E}\left[\frac{v_j^i(x)^4}{{\lambda_j^i}^2}\mathbb{1}\left(x_j\in C_i\right)\right]\\
&=a_1^2S_i(\lambda p_i)^2,
\end{aligned}\tag{C.49}$$

(e) using the relation $f_{i,\lambda} = \Sigma_{i,\lambda p_i}^{-1} \Sigma_{i,\overline{\lambda} p_i} f_{i,\overline{\lambda}}$.

Bound on $\mathbb{E}\left[\|\Delta_b\|_{\mathcal{H}}^2 \mid \{x_1, \dots x_n\}\right]$. We have

$$\begin{aligned}\mathbb{E}\left[\|\Delta_b\|_{\mathcal{H}}^2 \mid \{x_1, \dots x_n\}\right] &\overset{(a)}{\leq} 2\|f_{i,\lambda}\|_{\mathcal{H}}^2 + 2\mathbb{E}\left[\left\|\hat{f}_{i,\lambda}\right\|_{\mathcal{H}}^2 \mid \{x_1 \dots x_n\}\right] \\ &\overset{(b)}{\leq} 4\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{2}{\lambda}\frac{1}{n_i}\sum_{j=1}^{n}(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^2 \mathbb{1}\left(x_j \in C_i\right) + \frac{2\sigma^2}{\lambda}\end{aligned} \tag{C.50}$$

where we have (a) using $\|x+y\|_{\mathcal{H}}^2 \leq 2\|x\|_{\mathcal{H}}^2 + 2\|y\|_{\mathcal{H}}^2$, (b) using optimality of $\hat{f}_{i,\lambda}$ for the loss function in Eq. (3.4).

Overall Bound. Combining the above bounds with the terms in Eq. (C.47), we have

$$\frac{4}{n^2}\sum_{j=1}^{n}\mathbb{E}\left[\|w_j\|_{\mathcal{H}}^2\right] \leq \frac{4S_i(\lambda p_i)}{n}\left[4a_1\sqrt{p_i}A_i(\overline{\lambda})^2\right] + \frac{4\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2}{n}\left[4a_1^2 p_i \frac{(\lambda - \overline{\lambda})^2}{\lambda} S_i(\lambda p_i)^2 + 2\lambda p_i\right] \tag{C.51}$$

and

$$\begin{aligned}\mathbb{E}\left[\mathbb{E}\left[\|\Delta_b\|_{\mathcal{H}}^2 \mid x_1, \dots x_n\right]\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right] \leq &\left(4\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{2\sigma^2}{\lambda}\right)\mathbb{P}(\mathcal{E}_{cov}^c) \\ &+ \frac{2}{\lambda}\mathbb{E}\left[\frac{1}{n_i}\sum_{j=1}^{n}(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^2 \mathbb{1}\left(x_j \in C_i\right)\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right]\end{aligned} \tag{C.52}$$

Now,

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{n_i}(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^2 \mathbb{1}\left(x_j \in C_i\right) \mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right] \\
&\overset{(a)}{\leq} \sqrt{\mathbb{E}\left[\frac{1}{n_i^2}(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^4 \mathbb{1}\left(x_j \in C_i\right)\right]} \sqrt{\mathbb{P}(\mathcal{E}_{cov}^c)} \\
&\overset{(b)}{=} \sqrt{\mathbb{P}(\mathcal{E}_{cov}^c)} \sqrt{p_i} \sqrt{\mathbb{E}\left[(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^4 \,|\, x_j \in C_i\right] \mathbb{E}\left[\frac{1}{(1+Y)^2}\right]} \\
&\overset{(c)}{\leq} \sqrt{\mathbb{P}(\mathcal{E}_{cov}^c)} \sqrt{p_i} A_i(\overline{\lambda})^2 \sqrt{\left(\exp(-(n-1)p_i/8) + \frac{4}{((n-1)p_i)^2}\right)} \\
&\overset{(d)}{\leq} \frac{4}{n\sqrt{p_i}} \sqrt{\mathbb{P}(\mathcal{E}_{cov}^c)} A_i(\overline{\lambda})^2
\end{aligned} \tag{C.53}$$

where we have (a) using Cauchy-Schwarz, (b) using $n_i = \sum_{j=1}^n \mathbb{1}\left(x_j \in C_i\right)$, independence of $x_1, \ldots, x_n$, and letting $Y \sim Bin(n-1, p_i)$, (c) using Assumption 3.2 and $\mathbb{E}\left[\frac{1}{(1+Y)^2}\right] \leq \exp(-np/8) + \frac{4}{(np)^2}$ for $Y \sim Bin(n,p)$ with $p \leq 1/2$, (d) using $p_i \geq \frac{16\log(np_i/2)}{n-1}$.

Consequently, we have

$$\mathbb{E}\left[\mathbb{E}\left[\|\Delta_b\|_{\mathcal{H}}^2 \mid x_1, \ldots x_n\right] \mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right] \leq \left(4\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{2\sigma^2}{\lambda}\right)\mathbb{P}(\mathcal{E}_{cov}^c) + 8\sqrt{\mathbb{P}(\mathcal{E}_{cov}^c)}\frac{A_i(\overline{\lambda})^2}{\lambda\sqrt{p_i}} \tag{C.54}$$

Finally, plugging Eqs. (C.51) and (C.54) into Eq. (C.47) followed by Eq. (C.42), we have the bias bound

$$\begin{aligned}
&\|\mathbb{E}\left[\Delta_b\right]\|_{\Sigma_i}^2 \leq \\
&(CovErr_i(d, \lambda p_i, n))^2 \left(T_1 + T_2 + 2^{k+1}\left[\mathrm{CovErr}_i(d, \lambda p_i, n)\right]^k T_3 + 2^{k/2+3}\left[\mathrm{CovErr}_i(d, \lambda p_i, n)\right]^{k/2} T_4\right)
\end{aligned} \tag{C.55}$$

where we let

$$
\begin{aligned}
T_1 &= \frac{16a_1\sqrt{p_i}S_i(\lambda p_i)A_i(\overline{\lambda})^2}{n} \\
T_2 &= \left(\frac{16a_1^2(\lambda-\overline{\lambda})^2}{\lambda}\,\frac{p_iS_i(\lambda p_i)^2\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2}{n} + \frac{8\lambda p_i\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2}{n}\right) \\
T_3 &= \left(2\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{\lambda}\right)(\lambda_1^i + \lambda p_i) \\
T_4 &= \frac{(\lambda_1^i + \lambda p_i)A_i(\overline{\lambda})^2}{\lambda\sqrt{p_i}}
\end{aligned}
\tag{C.56}
$$

□

C.1.8 Variance Bound

In this section we provide a proof of the bound on $\mathbb{E}_D\left[\mathrm{Var}_i(\lambda, D)\right]$ in Lemma B.1. The bound with exact constants is stated below.

Lemma C.4. *Consider any $d \in \mathbb{N}, d \geq 1$, and $k \geq 2$. Suppose Assumption 3.1 holds for this k (with constant a_1), and Assumption 3.2 holds. Also, suppose $\forall i \in [m]$, p_i satisfies: $p_i = \Omega\left(\log n/n\right)$. Then we have*

$$
\begin{aligned}
\mathbb{E}\left[\mathit{Var}_i(\lambda, D)\right] \leq{}& \frac{4(\sigma^2 + a_1\sqrt{p_i}A_i(\overline{\lambda})^2)S_i(\lambda p_i)}{n} + 4\frac{(\overline{\lambda}-\lambda)^2p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 \\
&+ 2^{k+2}\left[\mathit{CovErr}_i(d, \lambda p_i, n)\right]^k W_1 + 2^{\frac{k}{2}+4}\left[\mathit{CovErr}_i(d, \lambda p_i, n)\right]^{k/2} W_2
\end{aligned}
\tag{C.57}
$$

where we let

$$
\begin{aligned}
W_1 &= \lambda_1^i\left(\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{2\lambda}\right) \\
W_2 &= \lambda_1^i\frac{A_i(\overline{\lambda})^2}{\lambda\sqrt{p_i}}
\end{aligned}
\tag{C.58}
$$

C.1.8.1 Proof of Lemma C.4

Proof. We want to bound the quantity $\mathbb{E}\left[\mathrm{Var}_i(\lambda, D)\right]$, where

$$\mathrm{Var}_i(\lambda, D) = \mathbb{E}\left[(\bar{f}_{i,\lambda}(x) - \hat{f}_{i,\lambda}(x))^2 \mathbb{1}\left(x \in C_i\right)\right] \tag{C.59}$$

$$= \left\|\bar{f}_{i,\lambda} - \hat{f}_{i,\lambda}\right\|_{\Sigma_i}^2 \tag{C.60}$$

Since $\bar{f}_{i,\lambda} = \mathbb{E}\left[\hat{f}_{i,\lambda}\right]$ minimizes $\mathbb{E}\left[\left\|\hat{f}_{i,\lambda} - f\right\|_{\Sigma_i}^2\right]$ for $f \in \mathcal{H}$, we can get:

$$\mathbb{E}\left[Var_i(\lambda, D)\right] = \mathbb{E}\left[\left\|\bar{f}_{i,\lambda} - \hat{f}_{i,\lambda}\right\|_{\Sigma_i}^2\right] \leq \mathbb{E}\left[\left\|f_{i,\overline{\lambda}} - \hat{f}_{i,\lambda}\right\|_{\Sigma_i}^2\right] \tag{C.61}$$

where $f_{i,\overline{\lambda}}$ is the solution of (3.7). Let $\Delta_v = \hat{f}_{i,\lambda} - f_{i,\overline{\lambda}}$.

Now, from first order optimality conditions for Eq (3.2), we have

$$(\hat{\Sigma}_i + \lambda p_i I)\hat{f}_{i,\lambda} = \widehat{\mathbb{E}}\left[y\phi_x \mathbb{1}\left(x \in C_i\right)\right] \tag{C.62}$$

$$= \widehat{\mathbb{E}}\left[f^*(x)\phi_x \mathbb{1}\left(x \in C_i\right)\right] + \widehat{\mathbb{E}}\left[\eta\phi_x \mathbb{1}\left(x \in C_i\right)\right] \tag{C.63}$$

Subtracting $(\hat{\Sigma} + \lambda p_i I) f_{i,\overline{\lambda}}$ from the above, we get,

$$(\hat{\Sigma}_i + \lambda p_i I)\Delta_v = \widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))\phi_x \mathbb{1}\left(x \in C_i\right) - \lambda p_i f_{i,\overline{\lambda}}\right] + \widehat{\mathbb{E}}\left[\eta\phi_x \mathbb{1}\left(x \in C_i\right)\right] \tag{C.64}$$

$$= \widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))\phi_x \mathbb{1}\left(x \in C_i\right) - \overline{\lambda} p_i f_{i,\overline{\lambda}}\right] + \widehat{\mathbb{E}}\left[\eta\phi_x \mathbb{1}\left(x \in C_i\right)\right] + (\overline{\lambda} - \lambda) p_i f_{i,\overline{\lambda}} \tag{C.65}$$

Thus,

$$\left(\Sigma_{i,\lambda p_i}^{-1/2}(\hat{\Sigma}_i - \Sigma_i)\Sigma_{i,\lambda p_i}^{-1/2} + I\right)\Sigma_{i,\lambda p_i}^{1/2}\Delta_v = \widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))\Sigma_{i,\lambda p_i}^{-1/2}\phi_x \mathbb{1}\left(x \in C_i\right) - \overline{\lambda} p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\overline{\lambda}}\right]$$
$$+ \widehat{\mathbb{E}}\left[\eta\Sigma_{i,\lambda p_i}^{-1/2}\phi_x \mathbb{1}\left(x \in C_i\right)\right] + (\overline{\lambda} - \lambda) p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\overline{\lambda}} \tag{C.66}$$

Let us define the event $\mathcal{E}_{cov} = \left\{ \left\| \Sigma_{i,\lambda p_i}^{-1/2} (\hat{\Sigma}_i - \Sigma_i) \Sigma_{i,\lambda p_i}^{-1/2} \right\| \leq 1/2 \right\}$. Note that from Corollary C.1, we have $\mathbb{P}\left(\mathcal{E}_{cov}^c\right) \leq 2^k \left[\text{CovErr}_i(d, \lambda p_i, n)\right]^k$. Now, under the event $\mathcal{E}_{cov}$,

$$\mathbb{E}\left[\left\| \Sigma_{i,\lambda p_i}^{1/2} \Delta_v \right\|_{\mathcal{H}}^2 \right] \leq 4\mathbb{E}\left[\left\| \widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\overline{\lambda}}(x)) \Sigma_{i,\lambda p_i}^{-1/2} \phi_x \mathbb{1}\left(x \in C_i\right) - \overline{\lambda} p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\overline{\lambda}} \right] \right\|_{\mathcal{H}}^2 \right] + 4\mathbb{E}\left[\left\| \widehat{\mathbb{E}}\left[\eta \Sigma_{i,\lambda p_i}^{-1/2} \phi_x \mathbb{1}\left(x \in C_i\right) \right] \right\|_{\mathcal{H}}^2 \right] + 4(\overline{\lambda} - \lambda)^2 p_i^2 \left\| \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\overline{\lambda}} \right\|_{\mathcal{H}}^2 \tag{C.67}$$

Now, we can control each of the component terms in the above inequality as follows:

$$\begin{aligned}
&4\mathbb{E}\left[\left\| \widehat{\mathbb{E}}\left[(f^*(x) - f_{i,\overline{\lambda}}(x)) \Sigma_{i,\lambda p_i}^{-1/2} \phi_x \mathbb{1}\left(x \in C_i\right) - \overline{\lambda} p_i \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\overline{\lambda}} \right] \right\|_{\mathcal{H}}^2 \right] \\
&\overset{(a)}{=} \frac{4}{n} \mathbb{E}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))^2 \left\| \Sigma_{i,\lambda p_i}^{-1/2} \phi_x \right\|_{\mathcal{H}}^2 \mathbb{1}\left(x \in C_i\right) \right] - \frac{4}{n} \overline{\lambda}^2 p_i^2 \left\| \Sigma_{i,\lambda p_i}^{-1/2} f_{i,\overline{\lambda}} \right\|_{\mathcal{H}}^2 \\
&\overset{(b)}{\leq} \frac{4}{n} \sqrt{\mathbb{E}\left[(f^*(x) - f_{i,\overline{\lambda}}(x))^4 \mathbb{1}\left(x \in C_i\right) \right]} \sqrt{\mathbb{E}\left[\left\| \Sigma_{i,\lambda p_i}^{-1/2} \phi_x \right\|_{\mathcal{H}}^4 \mathbb{1}\left(x \in C_i\right) \right]} \\
&\overset{(c)}{\leq} \frac{4}{n} a_1 \sqrt{p_i} A_i(\overline{\lambda})^2 S_i(\lambda p_i)
\end{aligned} \tag{C.68}$$

where we have (a) using independence of $\{x_1, \ldots, x_n\}$ and $\mathbb{E}\left[(f^*(x) - f_{i,\overline{\lambda}}(x)) \phi_x \mathbb{1}\left(x \in C_i\right) - \overline{\lambda} p_i f_{i,\overline{\lambda}} \right] = 0$ (via first order optimality conditions for $f_{i,\overline{\lambda}}$) , (b)using Cauchy-Schwarz and ignoring the negative quantity, (c) using Assumption 3.2 and $\mathbb{E}\left[\left\| \Sigma_{i,\lambda p_i}^{-1/2} \phi_x \right\|_{\mathcal{H}}^4 \mathbb{1}\left(x \in C_i\right) \right] \leq a_1^2 S_i(\lambda p_i)^2$ (via Assumption 3.1),

And,

$$
\begin{aligned}
&4\mathbb{E}\left[\left\|\widehat{\mathbb{E}}\left[\eta\Sigma_{i,\lambda p_i}^{-1/2}\phi_x\mathbb{1}\left(x\in C_i\right)\right]\right\|_{\mathcal{H}}^2\right]\\
&=4\mathbb{E}\left[\frac{1}{n^2}\sum_{j=1}^{n}\sum_{k=1}^{n}\eta_j\eta_k\left\langle\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\mathbb{1}\left(x_j\in C_i\right),\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_k}\mathbb{1}\left(x_k\in C_i\right)\right\rangle_{\mathcal{H}}\right]\\
&\overset{(a)}{=}4\mathbb{E}\left[\frac{1}{n^2}\sum_{j=1}^{n}\eta_j^2\left\langle\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j},\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\rangle_{\mathcal{H}}\mathbb{1}\left(x_j\in C_i\right)\right]\\
&\overset{(b)}{\leq}\frac{4\sigma^2S_i(\lambda p_i)}{n}
\end{aligned}
\tag{C.69}
$$

where we have (a) since $\mathbb{E}\left[\eta_j\eta_k\right]=0$ for $j\neq k$, (b) using $\mathbb{E}\left[\eta_j^2\right]\leq\sigma^2$, $\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{-1/2}\phi_{x_j}\right\|_{\mathcal{H}}^2\right]=S_i(\lambda p_i)$ and the independence of η_j and x_j,

And,

$$
\begin{aligned}
4(\overline{\lambda}-\lambda)^2p_i^2\left\|\Sigma_{i,\lambda p_i}^{-1/2}f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2&\leq 4(\overline{\lambda}-\lambda)^2p_i^2\frac{\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2}{\lambda_1^i+\lambda p_i}\\
&\leq 4\frac{(\overline{\lambda}-\lambda)^2p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2
\end{aligned}
\tag{C.70}
$$

Thus, overall, we have

$$
\begin{aligned}
\mathbb{E}\left[\left\|\Sigma_i^{1/2}\Delta_v\right\|_{\mathcal{H}}^2\right]&=\mathbb{E}\left[\left\|\Sigma_i^{1/2}\Delta_v\right\|_{\mathcal{H}}^2\mathbb{1}\left(\mathcal{E}_{cov}\right)\right]+\mathbb{E}\left[\left\|\Sigma_i^{1/2}\Delta_v\right\|_{\mathcal{H}}^2\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right]\\
&\leq\mathbb{E}\left[\left\|\Sigma_{i,\lambda p_i}^{1/2}\Delta_v\right\|_{\mathcal{H}}^2\mathbb{1}\left(\mathcal{E}_{cov}\right)\right]+\mathbb{E}\left[\left\|\Sigma_i^{1/2}\Delta_v\right\|_{\mathcal{H}}^2\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right]\\
&\leq\frac{4(\sigma^2+a_1\sqrt{p_i}A_i(\overline{\lambda})^2)S_i(\lambda p_i)}{n}+4\frac{(\overline{\lambda}-\lambda)^2p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2+\mathbb{E}\left[\left\|\Sigma_i^{1/2}\Delta_v\right\|_{\mathcal{H}}^2\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right]\\
&\leq\frac{4(\sigma^2+a_1\sqrt{p_i}A_i(\overline{\lambda})^2)S_i(\lambda p_i)}{n}+4\frac{(\overline{\lambda}-\lambda)^2p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2\\
&\quad+\lambda_1^i\mathbb{E}\left[\mathbb{E}\left[\left\|\Delta_v\right\|_{\mathcal{H}}^2\mid x_1\ldots x_n\right]\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right]
\end{aligned}
\tag{C.71}
$$

where in the last step, we use the fact that $\mathcal{E}_{cov}$ only depends on $\{x_1,\ldots x_n\}$.

Now, we have the following bound on $\mathbb{E}\left[\|\Delta_v\|_{\mathcal{H}}^2 \mid x_1 \dots x_n\right]$.

$$\begin{aligned}
\mathbb{E}\left[\|\Delta_v\|_{\mathcal{H}}^2 \mid x_1 \dots x_n\right] &= \mathbb{E}\left[\left\|\hat{f}_{i,\lambda} - f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 \mid x_1 \dots x_n\right] \\
&\leq 2\mathbb{E}\left[\left\|\hat{f}_{i,\lambda}\right\|_{\mathcal{H}}^2 \mid x_1 \dots x_n\right] + 2\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 \\
&\overset{(a)}{\leq} 4\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + 2\frac{\sigma^2}{\lambda} + \frac{2}{\lambda}\frac{1}{n_i}\sum_{j=1}^{n}(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^2 \mathbb{1}\left(x_j \in C_i\right)
\end{aligned} \tag{C.72}$$

where we have (a) using the optimality of $\hat{f}_{i,\lambda}$ in Eq. (3.4)

Plugging the above back into Eq. (C.71), we get

$$\begin{aligned}
\mathbb{E}\left[\mathrm{Var}_i(\lambda, D)\right] &\leq \frac{4(\sigma^2 + a_1\sqrt{p_i}A_i(\overline{\lambda})^2)S_i(\lambda p_i)}{n} + 4\frac{(\overline{\lambda}-\lambda)^2 p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + 4\lambda_1^i\mathbb{P}(\mathcal{E}_{cov}^c)\left(\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{2\lambda}\right) \\
&\quad + 4\frac{\lambda_1^i}{\lambda}\mathbb{E}\left[\frac{1}{n_i}\sum_{j=1}^{n}(f^*(x_j) - f_{i,\overline{\lambda}}(x_j))^2 \mathbb{1}\left(x_j \in C_i\right)\mathbb{1}\left(\mathcal{E}_{cov}^c\right)\right] \\
&\overset{(a)}{\leq} \frac{4(\sigma^2 + a_1\sqrt{p_i}A_i(\overline{\lambda})^2)S_i(\lambda p_i)}{n} + 4\frac{(\overline{\lambda}-\lambda)^2 p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + 4\lambda_1^i\mathbb{P}(\mathcal{E}_{cov}^c)\left(\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{2\lambda}\right) \\
&\quad + 16\frac{\lambda_1^i}{\lambda\sqrt{p_i}}\sqrt{\mathbb{P}(\mathcal{E}_{cov}^c)}A_i(\overline{\lambda})^2 \\
&\leq \frac{4(\sigma^2 + a_1\sqrt{p_i}A_i(\overline{\lambda})^2)S_i(\lambda p_i)}{n} + 4\frac{(\overline{\lambda}-\lambda)^2 p_i}{\lambda}\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 \\
&\quad + 2^{k+2}\lambda_1^i\left[\mathrm{CovErr}_i(d, \lambda p_i, n)\right]^k\left(\left\|f_{i,\overline{\lambda}}\right\|_{\mathcal{H}}^2 + \frac{\sigma^2}{2\lambda}\right) \\
&\quad + 2^{\frac{k}{2}+4}\lambda_1^i\left[\mathrm{CovErr}_i(d, \lambda p_i, n)\right]^{k/2}\frac{A_i(\overline{\lambda})^2}{\lambda\sqrt{p_i}}
\end{aligned} \tag{C.73}$$

where we have (a) using the same sequence of inequalities employed in Eq. (C.53). □

C.1.9 Proof of Lemma C.1

Proof. Using the triangle inequality, we obtain the decomposition

$$\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i-\Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k} \leq \underbrace{\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i-\hat{\Sigma}_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k}}_{T_1} \tag{C.74}$$
$$+\underbrace{\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d-\Sigma_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k}}_{T_2}+\underbrace{\left\|\Sigma_{i,\lambda}^{-1/2}(\Sigma_i^d-\Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\|}_{T_3}$$

Bound on T_1. Consider the term $\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i-\hat{\Sigma}_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|$. Using the definition of $\hat{\Sigma}_i$ and $\hat{\Sigma}_i^d$ from Eqs. (C.4) and (C.9), and then applying the triangle inequality, we have

$$\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i-\hat{\Sigma}_i^d)\Sigma_{i,\lambda}^{-1/2}\right\| \leq \frac{1}{n}\sum_{j=1}^{n}\left\|\Sigma_{i,\lambda}^{-1/2}((\phi_{x_j}\otimes\phi_{x_j})-(P_d\phi_{x_j}\otimes P_d\phi_{x_j}))\Sigma_{i,\lambda}^{-1/2}\right\|\mathbb{1}\left(x_j\in C_i\right) \tag{C.75}$$

Now, recall that for any $x\in\mathcal{X}$, we let $\Sigma_{i,\lambda}^{-1/2}\phi_x=\phi'_x$ and $P_d^{\perp}=\sum_{j>d}(v_j^i\otimes v_j^i)$. Also, $\phi'_x=P_d\phi'_x+P_d^{\perp}\phi'_x$. Then,

$$\begin{aligned}
&\left\|\Sigma_{i,\lambda}^{-1/2}((\phi_x\otimes\phi_x)-(P_d\phi_x\otimes P_d\phi_x))\Sigma_{i,\lambda}^{-1/2}\right\| \\
&=\|(\phi'_x\otimes\phi'_x)-(P_d\phi'_x\otimes P_d\phi'_x)\| \\
&=\left\|(P_d^{\perp}\phi'_x\otimes P_d^{\perp}\phi'_x)+(P_d^{\perp}\phi'_x\otimes P_d\phi'_x)+(P_d\phi'_x\otimes P_d^{\perp}\phi'_x)\right\| \\
&=\frac{1}{2}\left\|P_d^{\perp}\phi'_x\otimes(P_d^{\perp}\phi'_x+2P_d\phi'_x)+(P_d^{\perp}\phi'_x+2P_d\phi'_x)\otimes P_d^{\perp}\phi'_x\right\| \\
&\overset{(a)}{=}\frac{1}{2}\left(\left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}^2+\left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}\left\|P_d^{\perp}\phi'_x+2P_d\phi'_x\right\|_{\mathcal{H}}\right) \\
&\overset{(b)}{\leq}\left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}^2+\left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}\|P_d\phi'_x\|_{\mathcal{H}}
\end{aligned} \tag{C.76}$$

where we have (a) using $\|u \otimes v + v \otimes u\| = (\langle v, u \rangle_{\mathcal{H}} + \|u\|_{\mathcal{H}} \|v\|_{\mathcal{H}})$, and (b) using the triangle inequality.

Plugging this back into Eq. (C.75), we get

$$\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i - \hat{\Sigma}_i^d)\Sigma_{i,\lambda}^{-1/2}\right\| \leq \frac{1}{n}\sum_{j=1}^{n}\left(\left\|P_d^{\perp}\phi'_{x_j}\right\|_{\mathcal{H}}^2 + \left\|P_d^{\perp}\phi'_{x_j}\right\|_{\mathcal{H}}\left\|P_d\phi'_{x_j}\right\|_{\mathcal{H}}\right)\mathbb{1}\left(x_j \in C_i\right) \tag{C.77}$$

Taking expectation of the k^{th} power on both sides, and using the triangle inequality again, we get

$$\begin{aligned}
&\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i - \hat{\Sigma}_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k} \\
&\leq \frac{1}{n}\sum_{j=1}^{n}\mathbb{E}\left[\left\|P_d^{\perp}\phi'_{x_j}\right\|_{\mathcal{H}}^{2k}\mathbb{1}\left(x_j \in C_i\right)\right]^{1/k} + \frac{1}{n}\sum_{j=1}^{n}\mathbb{E}\left[\left\|P_d^{\perp}\phi'_{x_j}\right\|_{\mathcal{H}}^{k}\left\|P_d\phi'_{x_j}\right\|_{\mathcal{H}}^{k}\mathbb{1}\left(x_j \in C_i\right)\right]^{1/k} \\
&\overset{(a)}{\leq} \frac{1}{n}\sum_{j=1}^{n}\mathbb{E}\left[\left\|P_d^{\perp}\phi'_{x_j}\right\|_{\mathcal{H}}^{2k}\mathbb{1}\left(x_j \in C_i\right)\right]^{1/k} \\
&\qquad + \frac{1}{n}\sum_{j=1}^{n}\mathbb{E}\left[\left\|P_d^{\perp}\phi'_{x_j}\right\|_{\mathcal{H}}^{2k}\mathbb{1}\left(x_j \in C_i\right)\right]^{1/2k}\mathbb{E}\left[\left\|P_d\phi'_{x_j}\right\|_{\mathcal{H}}^{2k}\mathbb{1}\left(x_j \in C_i\right)\right]^{1/2k}
\end{aligned} \tag{C.78}$$

where we have (a) using the Cauchy-Schwarz inequality.

Now, as a consequence of the reproducing property of kernels, we note that ϕ_x, for any $x \in \mathcal{X}$, has the representation:

$$\phi_x = \sum_j v_j^i(x)v_j^i \tag{C.79}$$

Thus,

$$\begin{aligned}
\phi'_x = \Sigma_{i,\lambda}^{-1/2}\phi_x &= \sum_j \frac{v_j^i(x)}{\sqrt{\lambda_j^i+\lambda}} v_j^i \\
\Rightarrow\ P_d^{\perp}\phi'_x &= \sum_{j>d} \frac{v_j^i(x)}{\sqrt{\lambda_j^i+\lambda}} v_j^i \\
\Rightarrow\ \left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}^2 &= \sum_{j>d} \frac{(v_j^i(x))^2}{\lambda_j^i+\lambda} \\
\Rightarrow\ \left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}^{2k} &= \left(\sum_{j>d} \frac{(v_j^i(x))^2}{\lambda_j^i+\lambda}\right)^k \\
&= \left((\sum_{j>d} \lambda_j^i/(\lambda_j^i+\lambda)) \sum_{j>d} \frac{\lambda_j^i/(\lambda_j^i+\lambda)}{(\sum_{j>d}\lambda_j^i/(\lambda_j^i+\lambda))} (v_j^i(x))^2/\lambda_j^i\right)^k \\
&\overset{(a)}{\leq} \left(\sum_{j>d} \frac{\lambda_j^i}{\lambda_j^i+\lambda}\right)^k \left(\sum_{j>d} \frac{\lambda_j^i/(\lambda_j^i+\lambda)}{(\sum_{j>d}\lambda_j^i/(\lambda_j^i+\lambda))} \left(\frac{(v_j^i(x))^2}{\lambda_j^i}\right)^k\right)
\end{aligned} \tag{C.80}$$

where we have (a) using Jensen's inequality.

Therefore, using Assumption 3.1, we get

$$\mathbb{E}\left[\left\|P_d^{\perp}\phi'_x\right\|_{\mathcal{H}}^{2k} \mathbb{1}\left(x \in C_i\right)\right]^{1/k} \leq a_1 \left(\sum_{j>d} \frac{\lambda_j^i}{\lambda_j^i+\lambda}\right) \tag{C.81}$$

Similarly, we can obtain

$$\mathbb{E}\left[\left\|P_d\phi'_x\right\|_{\mathcal{H}}^{2k} \mathbb{1}\left(x \in C_i\right)\right]^{1/k} \leq a_1 \left(\sum_{j=1}^{d} \frac{\lambda_j^i}{\lambda_j^i+\lambda}\right) \tag{C.82}$$

Combining these bounds gives

$$\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i - \hat{\Sigma}_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k} \leq a_1\left(\sum_{j>d}\frac{\lambda_j^i}{\lambda_j^i+\lambda} + \sqrt{\sum_{j>d}\frac{\lambda_j^i}{\lambda_j^i+\lambda}}\sqrt{\sum_{j=1}^{d}\frac{\lambda_j^i}{\lambda_j^i+\lambda}}\right) \tag{C.83}$$
$$= a_1\left(L_i(d,\lambda) + \sqrt{L_i(d,\lambda)U_i(d,\lambda)}\right)$$

where $L_i(d,\lambda) = \sum_{j>d}\frac{\lambda_j^i}{\lambda_j^i+\lambda}$ and $U_i(d,\lambda) = \sum_{j=1}^{d}\frac{\lambda_j^i}{\lambda_j^i+\lambda}$.

Bound on T_2. We want to bound the quantity $\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d - \Sigma_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k}$. Using the definition of $\hat{\Sigma}_i^d$ from Eq. (C.9), we have

$$\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d - \Sigma_i^d)\Sigma_{i,\lambda}^{-1/2} = \frac{1}{n}\sum_{j=1}^{n}\left(\Sigma_{i,\lambda}^{-1/2}(P_d\phi_{x_j}\otimes P_d\phi_{x_j}\mathbb{1}\left(x_j\in C_i\right))\Sigma_{i,\lambda}^{-1/2} - \Sigma_{i,\lambda}^{-1/2}\Sigma_i^d\Sigma_{i,\lambda}^{-1/2}\right)$$
$$= \frac{1}{n}\sum_{j=1}^{n}\left((P_d\phi'_{x_j}\otimes P_d\phi'_{x_j})\mathbb{1}\left(x_j\in C_i\right) - \Sigma_{i,\lambda}^{-1/2}\Sigma_i^d\Sigma_{i,\lambda}^{-1/2}\right) \tag{C.84}$$

where $\phi'_x = \Sigma_{i,\lambda}^{-1/2}\phi_x$, for any $x\in\mathcal{X}$. Now, as seen in Eq. C.80, we have the representation:

$$P_d\phi'_{x_j} = \sum_{m=1}^{d}\frac{v_m^i(x_j)}{\sqrt{\lambda_m^i+\lambda}}v_m^i \tag{C.85}$$

$$\Rightarrow\ P_d\phi'_{x_j}\otimes P_d\phi'_{x_j} = \sum_{m=1}^{d}\sum_{n=1}^{d}\frac{v_m^i(x_j)v_n^i(x_j)}{\sqrt{\lambda_m^i+\lambda}\sqrt{\lambda_n^i+\lambda}}(v_m^i\otimes v_n^i) \tag{C.86}$$

Also, using the definition of Σ_i^d from Eq. C.10, we have the relation:

$$\Sigma_{i,\lambda}^{-1/2}\Sigma_i^d\Sigma_{i,\lambda}^{-1/2} = \sum_{m=1}^{d}\frac{\lambda_m^i}{\lambda_m^i+\lambda}(v_m^i\otimes v_m^i) \tag{C.87}$$

Now, let $A_j \in \mathbb{R}^{d\times d}$ be a matrix such that

$$\text{For } m \neq n,\ A_j(m,n) = v_m^i(x_j)v_n^i(x_j)\mathbb{1}\left(x_j \in C_i\right)/\sqrt{(\lambda_m^i+\lambda)(\lambda_n^i+\lambda)} \quad \text{(C.88)}$$

$$A_j(m,m) = \left(v_m^i(x_j)^2\mathbb{1}\left(x_j \in C_i\right) - \lambda_m^i\right)/(\lambda_m^i+\lambda) \quad \text{(C.89)}$$

Also, let $B = \sum_{j=1}^n A_j/n$. Then,

$$\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d - \Sigma_i^d)\Sigma_{i,\lambda}^{-1/2} = \frac{1}{n}\sum_{j=1}^n\left(\sum_{m=1}^d\sum_{n=1}^d A_j(m,n)(v_m^i \otimes v_n^i)\right) \quad \text{(C.90)}$$

$$= \sum_{m=1}^d\sum_{n=1}^d B(m,n)(v_m^i \otimes v_n^i) \quad \text{(C.91)}$$

So, we get

$$\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d - \Sigma_i^d)\Sigma_{i,\lambda}^{-1/2}\right\| = \left\|\sum_{m=1}^d\sum_{n=1}^d B(m,n)(v_m^i \otimes v_n^i)\right\| = \|B\|_2 = \left\|\frac{1}{n}\sum_{j=1}^n A_j\right\|_2 \quad \text{(C.92)}$$

where $\|\cdot\|_2$ corresponds to the usual spectral norm for finite dimensional matrices.

Thus to bound $\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d - \Sigma_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k}$, we need to bound $\mathbb{E}\left[\left\|\frac{1}{n}\sum_{j=1}^n A_j\right\|_2^k\right]^{1/k}$. To do this, we can use the following result from [12] (similar to its use in [67]) which provides a bound on the moment of the spectral norm of a sum of finite dimensional random matrices.

Lemma C.5. *Theorem A.1 [12] Let $q \geq 2$, and fix $r \geq \max\{q, \log d\}$. Consider a finite sequence $\{Y_i\}$ of independent, symmetric, random, self-adjoint matrices with dimension $d \times d$. Then,*

$$\mathbb{E}\left[\left\|\sum_i Y_i\right\|_2^q\right]^{1/q} \leq \sqrt{er}\left\|\sum_i \mathbb{E}\left[Y_i^2\right]\right\|_2^{1/2} + 2er\,\mathbb{E}\left[\max_i \|Y_i\|_2^q\right]^{1/q} \quad \text{(C.93)}$$

We apply Lemma C.5 in our case with the sequence of matrices $\left\{\frac{A_j}{n}\right\}$ to get

$$\mathbb{E}\left[\left\|\frac{1}{n}\sum_{j=1}^{n} A_j\right\|_2^k\right]^{1/k} \leq \frac{\sqrt{e\log d}}{n}\left\|\sum_{j=1}^{n}\mathbb{E}\left[A_j^2\right]\right\|_2^{1/2} + \frac{2e\log d}{n}\,\mathbb{E}\left[\max_j \|A_j\|_2^k\right]^{1/k} \tag{C.94}$$

Now, we can bound $\left\|\sum_{j=1}^{n}\mathbb{E}\left[A_j^2\right]\right\|_2$ as:

$$\begin{aligned}
\left\|\sum_{j=1}^{n}\mathbb{E}\left[A_j^2\right]\right\|_2 &\overset{(a)}{\leq} \sum_{j=1}^{n}\left\|\mathbb{E}\left[A_j^2\right]\right\|_2 \\
&\overset{(b)}{\leq} \sum_{j=1}^{n}\mathbb{E}\left[\|A_j\|_2^2\right] \\
&\overset{(c)}{\leq} \sum_{j=1}^{n}\mathbb{E}\left[\mathrm{Tr}\left(A_j\right)^2\right] \\
&= \sum_{j=1}^{n}\mathbb{E}\left[\left(\sum_{m=1}^{d}\frac{v_m^i(x_j)^2\mathbb{1}\left(x_j\in C_i\right)}{\lambda_m^i+\lambda}\right)^2\right] + \sum_{j=1}^{n}\left(\sum_{m=1}^{d}\frac{\lambda_m^i}{\lambda_m^i+\lambda}\right)^2 \\
&\quad - \sum_{j=1}^{n}2\left(\sum_{m=1}^{d}\frac{\lambda_m^i}{\lambda_m^i+\lambda}\right)\mathbb{E}\left[\left(\sum_{m=1}^{d}\frac{v_m^i(x_j)^2\mathbb{1}\left(x_j\in C_i\right)}{\lambda_m^i+\lambda}\right)\right] \\
&\overset{(d)}{=} \sum_{j=1}^{n}\mathbb{E}\left[\left(\sum_{m=1}^{d}\frac{v_m^i(x_j)^2\mathbb{1}\left(x_j\in C_i\right)}{\lambda_m^i+\lambda}\right)^2\right] - \sum_{j=1}^{n}\left(\sum_{m=1}^{d}\frac{\lambda_m^i}{\lambda_m^i+\lambda}\right)^2 \\
&\leq \sum_{j=1}^{n}\mathbb{E}\left[\left(\sum_{m=1}^{d}\frac{v_m^i(x_j)^2\mathbb{1}\left(x_j\in C_i\right)}{\lambda_m^i+\lambda}\right)^2\right] \\
&= \sum_{j=1}^{n}\left(\sum_{m=1}^{d}\lambda_m^i/(\lambda_m^i+\lambda)\right)^2\mathbb{E}\left[\left(\sum_{m=1}^{d}\frac{\lambda_m^i/(\lambda_m^i+\lambda)}{\sum_{m=1}^{d}\lambda_m^i/(\lambda_m^i+\lambda)}\frac{v_m^i(x_j)^2\mathbb{1}\left(x_j\in C_i\right)}{\lambda_m^i}\right)^2\right] \\
&\overset{(e)}{\leq} \sum_{j=1}^{n}U_i(d,\lambda)^2\,\mathbb{E}\left[\sum_{m=1}^{d}\frac{\lambda_m^i/(\lambda_m^i+\lambda)}{\sum_{m=1}^{d}\lambda_m^i/(\lambda_m^i+\lambda)}\left(\frac{v_m^i(x_j)^2\mathbb{1}\left(x_j\in C_i\right)}{\lambda_m^i}\right)^2\right] \\
&\overset{(f)}{\leq} \sum_{j=1}^{n}U_i(d,\lambda)^2a_1^2 \\
&= n\,U_i(d,\lambda)^2a_1^2 \qquad \text{(C.95)}
\end{aligned}$$

where we have (a) using the triangle inequality, (b) using Jensen's inequality, (c) since the spectral norm is upper bounded by the trace, (d) using the fact

that $\mathbb{E}\left[v_m^i(x_j)^2 \mathbb{1}\left(x_j \in C_i\right)\right] = \lambda_m^i$ for any m, (e) using Jensen's inequality again, and (f) using Assumption 3.1.

We can also bound $\mathbb{E}\left[\max_j \|A_j\|_2^k\right]$ as:

$$\begin{aligned}
\mathbb{E}\left[\max_j \|A_j\|_2^k\right] &\leq \sum_{j=1}^{n} \mathbb{E}\left[\|A_j\|_2^k\right] \\
&\overset{(a)}{\leq} \sum_{j=1}^{n} \mathbb{E}\left[\left(\sum_{m=1}^{d} \frac{v_m^i(x_j)^2 \mathbb{1}\left(x_j \in C_i\right)}{\lambda_m^i + \lambda} + \frac{\lambda_1^i}{\lambda_1^i + \lambda}\right)^k\right] \\
&\overset{(b)}{\leq} \sum_{j=1}^{n} \mathbb{E}\left[2^k \left(\sum_{m=1}^{d} \frac{v_m^i(x_j)^2 \mathbb{1}\left(x_j \in C_i\right)}{\lambda_m^i + \lambda}\right)^k + 2^k \left(\frac{\lambda_1^i}{\lambda_1^i + \lambda}\right)^k\right] \\
&\overset{(c)}{\leq} n\, 2^k \left(U_i(d,\lambda)^k a_1^k + \left(\frac{\lambda_1^i}{\lambda_1^i + \lambda}\right)^k\right) \qquad \text{(C.96)}
\end{aligned}$$

where we have (a) using the triangle inequality for the spectral norm and the fact that $A_j = vv^T - D$ with $v = \left\{v_m^i(x_j)\mathbb{1}\left(x_j \in C_i\right)/\sqrt{\lambda_m^i + \lambda}\right\}_{m=1}^{d}$ and $D = \operatorname{diag}\left(\{\lambda_m^i/(\lambda_m^i + \lambda)\}_{m=1}^{d}\right)$, (b) using the inequality $(a+b)^k \leq 2^k(a^k + b^k)$, and (c) using Jensen's inequality and Assumption 3.1.

Thus,

$$\mathbb{E}\left[\max_j \|A_j\|_2^k\right]^{1/k} \leq 2n^{1/k}\left(U_i(d,\lambda)a_1 + \left(\frac{\lambda_1^i}{\lambda_1^i + \lambda}\right)\right) \qquad \text{(C.97)}$$

Plugging these bounds into Eq. (C.92), we finally have

$$\begin{aligned}
\mathbb{E}\left[\left\|\Sigma_{i,\lambda}^{-1/2}(\hat{\Sigma}_i^d - \Sigma_i^d)\Sigma_{i,\lambda}^{-1/2}\right\|^k\right]^{1/k} &= \mathbb{E}\left[\left\|\frac{1}{n}\sum_{j=1}^{n} A_j\right\|_2^k\right]^{1/k} \\
&\leq a_1\sqrt{\frac{e\log d}{n}}\, U_i(d,\lambda) + \frac{4e\log d}{n^{1-1/k}}\left(a_1 U_i(d,\lambda) + \frac{\lambda_1^i}{\lambda_1^i + \lambda}\right) \qquad \text{(C.98)}
\end{aligned}$$

Bound on T_3. We wish to bound $\left\|\Sigma_{i,\lambda}^{-1/2}(\Sigma_i^d - \Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\|$. Using the definition of Σ_i^d from Eq. (C.10), we can get

$$\Sigma_{i,\lambda}^{-1/2}(\Sigma_i^d - \Sigma_i)\Sigma_{i,\lambda}^{-1/2} = -\sum_{j>d} \frac{\lambda_j^i}{\lambda_j^i + \lambda}(v_j^i \otimes v_j^i) \tag{C.99}$$

Thus,

$$\left\|\Sigma_{i,\lambda}^{-1/2}(\Sigma_i^d - \Sigma_i)\Sigma_{i,\lambda}^{-1/2}\right\| = \frac{\lambda_{d+1}^i}{\lambda_{d+1}^i + \lambda} \tag{C.100}$$

Overall Bound. Combining the bounds on the terms T_1, T_2 and T_3, we get the final bound in the lemma. □

Appendix D

Appendix C - Proofs for Chapter 4

D.1 Proof of Lemma 4.1

By Condition 1, we know that for any $I \subseteq [n]$, $|I| = n - s$, we have $\mathbf{1} \in \text{span}\{b_i \,|\, i \in I\}$. In other words, there exists at least one $x \in \mathbb{R}^{(n-s)}$ such that:

$$xB(I,:) = \mathbf{1} \tag{D.1}$$

Therefore, by construction, we have: $AB = \mathbf{1}_{\binom{n}{s} \times n}$, and the scheme (A, B) is robust to **any** s stragglers.

D.1.1 Proof of Theorem 4.1

Consider any scheme (A, B) robust to **any** s stragglers, with $B \in \mathbb{R}^{n \times k}$. Now, construct a bipartite graph between n workers, $\{W_1, \ldots, W_n\}$, and k partitions, $\{P_1, \ldots, P_k\}$, where we add an edge (i, j) if worker i and partition j is worker i has access to partition j. In other words, for any $i \in [n], j \in [k]$:

$$e_{ij} = \begin{cases} 1 & \text{if } B(i,j) \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{D.2}$$

Now, it is easy to see that the degree of the i^{th} worker W_i is $\|b_i\|_0$.

Also, for any partition P_j, its degree must be at least $(s+1)$. If its degree is s or less, then consider the scenario where all its neighbors are stragglers. In

this case, there is no non-straggler worker with access to P_j, which contradicts robustness to **any** s stragglers.

Based on the above discussion, and using the fact that the sum of degrees of the workers in the bipartite graph must be the same as the sum of degrees of partitions, we get:

$$\sum_{i=1}^{n} \|b_i\|_0 \geq k(s+1) \tag{D.3}$$

Since we assume all workers get access to the same number of partitions, this gives:

$$\|b_i\|_0 \geq \frac{k(s+1)}{n}, \text{ for any } i \in [n] \tag{D.4}$$

D.2 Proof of Theorem 4.2

Consider groups of partitions $\{G_1, \ldots, G_{n/(s+1)}\}$ as follows:

$$G_1 = \{P_1, \ldots, P_{s+1}\}$$

$$G_2 = \{P_{s+2}, \ldots, P_{2s+2}\}$$

$$\vdots \tag{D.5}$$

$$G_{n/(s+1)} = \{P_{n-s}, \ldots, P_n\} \tag{D.6}$$

Fix some set $I \subseteq [n], |I| = n - s$. Based on our construction, it is easy to observe that for any group G_j, there exists some index in I, say $i_{G_j} \in I$, such that the corresponding row in B, $b_{i_{G_j}}$ has all 1s at partitions in G_j and 0s elsewhere. This is because there are $(s+1)$ rows of B that correspond in

this way to G_j (one in each block $\overline{B}_{\text{block}}$), and so at least one would survive in the set I of cardinality $(n-s)$. Now, it is trivial to see that:

$$\mathbf{1} \in \text{span}\{b_{i_{G_j}} \,|\, j = 1, \ldots, n/(s+1)\} \tag{D.7}$$

Also, since

$$\text{span}\{b_{i_{G_j}} \,|\, j = 1, \ldots, n/(s+1)\} \subseteq \text{span}\{b_i \,|\, i \in I\}, \tag{D.8}$$

we have $\mathbf{1} \in \text{span}\{b_i \,|\, i \in I\}$.

Finally, since the above holds for any set I, we get that B satisfies Condition 1. The remainder of the theorem follows from Lemma 4.1.

D.3 Proof of Theorem 4.3

Consider the subspace given by the null space of the random matrix H (constructed in Algorithm 4.2):

$$S = \{x \in \mathbb{R}^n \,|\, Hx = 0\} \tag{D.9}$$

Note that H has $(n-1)s$ different random values (s for each column), since its last column is simply the negative sum of its previous $(n-1)$ columns. Now, we have the following Lemma listing some properties of H and S.

Lemma D.1. *Consider $H \in \mathbb{R}^{ss \times n}$ as constructed in Algorithm 4.2, and the subspace S as defined in Eq. D.9. Then, the following hold:*

- *Any s columns of H are linearly independent with probability* 1

- $dim(S) = n - s$ *with probability* 1
- $\mathbf{1} \in S$, *where* $\mathbf{1}$ *is the all-ones vector*

For $i \in [n]$, let S_i denote the set $S_i = \{i \mod n, (i+1) \mod n, \ldots, (i+s) \mod n\}$. Then, S_i corresponds to the support of the i^{th} row of B in our construction, as also given by the support structure in Eq. (4.10).

Recall that we denote the i^{th} row of B by b_i. By our construction, we have:

$$b_i(i) = 1$$
$$b_i(S_i \setminus \{i\}) = -H^{-1}_{S_i \setminus \{i\}} H_i \tag{D.10}$$

Now, we have the following lemma;

Lemma D.2. *Consider the* i^{th} *row of* B *constructed using Algorithm 4.2 (also shown in Eq. D.10). Then,*

- $b_i \in S$
- *Every element of* $b_i(S_i \setminus \{i\})$ *is non-zero with probability 1*
- *For any subset* $I \subseteq [n]$, $|I| = n - s$, *the set of vectors* $\{b_i \,|\, i \in I\}$ *is linearly independent with probability 1*

Now, using Lemma D.2, we can conclude that for any subset $I \subseteq [n]$, $|I| = n - s$, $dim\left(\text{span}\{b_i \,|\, i \in I\}\right) = n - s$ and $\text{span}\{b_i \,|\, i \in I\} \subseteq S$.

Consequently, from Lemma D.1, since $dim(S) = n - s$ and $\mathbf{1} \in S$, this implies that:

$$\text{span}\{b_i \,|\, i \in I\} = S \text{ with probability 1} \tag{D.11}$$

and, $\mathbf{1} \in \text{span}\{b_i \,|\, i \in I\}$. Taking union bound over every I shows that B satisfies Condition 1. The remainder of the theorem follows from Lemma 4.1.

D.3.1 Proof of Lemma D.1

Consider any subset $I \subseteq n$, $|I| = s$ such that $n \notin I$. Then, all the elements of H_I are independent, and $det(H_I)$ is a polynomial in the elements of H_I. Consequently, since every element is drawn from a continuous probability distribution (in particular, Gaussian), the set $\{H_I \,|\, det(H_I) = 0\}$ is a zero measure set. So, $P\left(det(H_I) \neq 0\right) = 1$, and thus the columns of H_I are linearly independent with probability 1.

If $n \in I$, then we have:

$$det(H_I) = det(\widetilde{H}) \tag{D.12}$$

where we let $\widetilde{H} = \left[H_{I\setminus\{n\}}, -\sum_{i\in[n]\setminus I} H_i\right]$. The elements of $\widetilde{H}$ are independent, so using the same argument as above, we again have $P(det(H_I) = det(\widetilde{H}) \neq 0) = 1$. Finally, taking a union bound over all sets I of cardinality s shows that any s columns of H are linearly independent.

Since any s columns in H are linearly independent, this implies that $rank(H) = s$. Since the subspace S is simply the null space of H, we have $dim(S) = n - s$.

Finally, since $H_n = -\sum_{i\in[n-1]} H_i$ (by construction), we have $H\mathbf{1} = 0$ and thus $\mathbf{1} \in S$.

D.3.2 Proof of Lemma D.2

By construction of b_i, we have:

$$Hb_i = H_i + H_{S_i\setminus\{i\}} b_i(S_i \setminus \{i\}) = H_i - H_i = 0 \tag{D.13}$$

Thus, $b_i \in S$.

Now, if possible, let for some $k \in S_i \setminus \{i\}$, $b_i(k) = 0$. Then, since $b_i \in S$, we have:

$$Hb_i = H_i + H_{S_i\setminus\{i,k\}} b_i(S_i \setminus \{i,k\}) = 0 \tag{D.14}$$

Consequently, the set of columns $\{j \,|\, j \in S_i \setminus \{i,k\}\} \cup \{i\}$ is linearly dependent which contradicts H having any s columns being linearly independent (in Lemma D.1). Therefore, we must have every element of $b_i(S_i \setminus \{i\})$ being non-zero.

Now, consider any subset $I \subseteq [n], |I| = n - s$. We shall show that the matrix B_I (corresponding to the rows of B with indices in I) has rank $n - s$ with probability 1. Consequently, the set of vectors $\{b_i \,|\, i \in I\}$ would be linearly independent. To show this, we consider some $n - s$ columns of B_I, say given by the set $J \subseteq [n], |J| = n - s$, and denote the sub-matrix of columns by $B_{I,J}$. Then, it suffices to show that $\det(B_{I,J}) \neq 0$. Now, by the construction in Algorithm 4.2, we have: $\det(B_{I,J}) = \text{poly}_1(H)/\text{poly}_2(H)$, for some polynomials $\text{poly}_1(\cdot)$ and $\text{poly}_2(\cdot)$ in the entries of H. Therefore, if we can

show that there exists at least one H' with $H'\mathbf{1} = \mathbf{0}$ and $\text{poly}_1(H')/\text{poly}_2(H') \neq 0$, then under a choice of i.i.d. standard Gaussian entries of H, we would have:

$$\mathbb{P}\left(\text{poly}_1(H)/\text{poly}_2(H) \neq 0\right) = 1 \tag{D.15}$$

The remainder of this proof is dedicated to showing that such an H' exists. To show this, we shall consider a matrix $\widetilde{B} \in \mathbb{R}^{n-s\times n}$ such that $\text{supp}(\widetilde{B}) = \text{supp}(B_I)$ and $\det(\widetilde{B}_{:,J}) \neq 0$, where $\widetilde{B}_{:,J}$ corresponds to the sub-matrix of $\widetilde{B}$ with columns in the set J. Given such a $\widetilde{B}$, we shall show that there exists an $s \times n$ matrix H' (with $H'\mathbf{1} = \mathbf{0}$) such that when we run Algorithm 4.2 with this H', we get a matrix B' s.t. $B'_I = \widetilde{B}$ *i.e.* the output matrix from Algorithm 4.2 is identical to our random choice $\widetilde{B}$ on the rows in the set I. This suffices to show the existence of an H' such that $\text{poly}_1(H')/\text{poly}_2(H') \neq 0$, since $\text{poly}_1(H')/\text{poly}_2(H') = \det(B'_{I,J}) = \det(\widetilde{B}_J) \neq 0$.

Let us pick a random matrix $\widetilde{B}$ as:

$$\widetilde{B} = B^r_I D \tag{D.16}$$

where B^r_I is a matrix with the same support as B_I and with each non-zero entry i.i.d. standard Gaussian, and D is a diagonal matrix such that $D_{ii} = \sum_{j=1}^{n-s} B^r_I(j,i)$, $i \in [n]$. Note that a consequence of the above choice of $\widetilde{B}$ is that the sum of all its rows is the all $\mathbf{1}$s vector. Now, it can be shown that any $(n-s)$ columns of $\widetilde{B}$ form an invertible sub-matrix with probability 1. Let S_i be the support of the i^{th} row of B. The rows of B^r_I have the supports $S_i, i \in I$. Now because of the cyclic support structure in B, any collection

$\{i_1, i_2, \ldots, i_k\} (0 \leq k \leq n-s)$ satisfies the property:

$$|\cup_{j=1}^{k} S_{i_j}| \geq s + k \tag{D.17}$$

Using Lemma 4 in [17], this implies that there is a perfect matching between the rows of B_I^r and any of its $(n-s)$ columns . Consequently, with probability 1, any $(n-s)$ columns of B_I^r form an invertible sub-matrix. Also, since every column of B_I^r contains at least one non-zero (again, owing to the support structure of B), this implies that with probability 1, all the diagonal entries of D are non-zero. Combining the above two observations, we can infer that any $(n-s)$ columns of $\widetilde{B}$ form an invertible sub-matrix with probability 1.

So far, we have shown existence of a matrix $\widetilde{B}$ with the following properties: (i) $\widetilde{B}$ has the same support structure as B_I, (ii) any $(n-s)$ columns of $\widetilde{B}$ form invertible sub-matrix, (iii) the sum of all rows of $\widetilde{B}$ is the all **1**s vector. Now, for any such $\widetilde{B}$, we shall show that there exists an H' such that $H'\widetilde{B}^T = \mathbf{0}$ such that any s columns of H' form an invertible sub-matrix. This implies that when we run Algorithm 4.2 with this H', the output matrix would be the same as $\widetilde{B}$ on the rows in the set I. The remainder of the proof then follows from our earlier discussion.

Now, consider any set $Q \subseteq [n], |Q| \leq s$. Suppose we pick any invertible $H'_{:,Q}$, and set $H'_{:,[n]\setminus Q} = -H'_{:,Q}\widetilde{B}^T_{:,Q}(\widetilde{B}^T_{:,[n]\setminus Q})^{-1}$. Then, such an H' satisfies $H'\widetilde{B}^T = 0$ and its columns in the set Q form an invertible sub-matrix. Now, since invertibility on the set Q simply corresponds to $\det(H'_{:,Q}) \neq 0$ (*i.e.* some fixed polynomial being non-zero), if we actually picked a uniformly random

H' on the subspace $H'\widetilde{B}^T = 0$, then

$$\mathbb{P}\left(\det(H'_{:,Q}) \neq 0 \,|\, H'\widetilde{B}^T = 0\right) = 1 \tag{D.18}$$

Taking a union bound over all Qs, we get that

$$\mathbb{P}\left(\text{any } s \text{ columns of } H' \text{ form an invertible sub-matrix} \,|\, H'\widetilde{B}^T = 0\right) = 1 \tag{D.19}$$

Thus, there exists an H' satisfying $H'\widetilde{B}^T = 0$ with any s of its columns forming an invertible sub-matrix. Also, since the sum of all rows of $\widetilde{B}$ is $\mathbf{1}$, this implies $H'\mathbf{1} = \mathbf{0}$.

Bibliography

[1] P. Abbeel, D. Koller, and A. Y. Ng. Learning factor graphs in polynomial time and sample complexity. *Jour. Mach. Learning Res.*, 7:1743–1788, 2006.

[2] A. Alaoui and M. W. Mahoney. Fast randomized kernel ridge regression with statistical guarantees. In *Advances in Neural Information Processing Systems 28*, pages 775–783. Curran Associates, Inc., 2015.

[3] A. Anandkumar, V. Y. F. Tan, and A.S. Willsky. High-Dimensional Structure Learning of Ising Models : Local Separation Criterion. *Preprint*, June 2011.

[4] F. R. Bach. Sharp analysis of low-rank kernel matrix approximations. In *COLT 2013 - The 26th Annual Conference on Learning Theory, June 12-14, 2013, Princeton University, NJ, USA*, pages 185–209, 2013.

[5] O. Bannerjee, , L. El Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Jour. Mach. Lear. Res.*, 9:485–516, March 2008.

[6] A.L. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[7] J. Besag. Statistical analysis of non-lattice data. *The Statistician*, 24(3):179–195, 1975.

[8] L. Bottou and V. N. Vapnik. Local learning algorithms. *Neural Computation*, 4(6):888–900, 1992.

[9] Guy Bresler, Elchanan Mossel, and Allan Sly. Reconstruction of markov random fields from samples: Some observations and algorithms. In *Proceedings of the 11th international workshop, APPROX 2008, and 12th international workshop, RANDOM 2008 on Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques*, APPROX '08 / RANDOM '08, pages 343–356. Springer-Verlag, 2008.

[10] A. Caponnetto and E. De Vito. Optimal rates for the regularized least-squares algorithm. *Found. Comput. Math.*, 7(3):331–368, July 2007.

[11] J. Chen, R. Monga, S. Bengio, and R. Jozefowicz. Revisiting Distributed Synchronous SGD. *ArXiv e-prints*, April 2016.

[12] R. Y. Chen, A. Gittens, and J. A. Tropp. The masked sample covariance estimator: an analysis using matrix concentration inequalities. *Information and Inference*, 2012.

[13] I. Csiszár and Z. Talata. Consistent estimation of the basic neighborhood structure of Markov random fields. *The Annals of Statistics*, 34(1):123–145, 2006.

[14] F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2002.

[15] B. Dai, B. Xie, N. He, Y. Liang, A. Raj, M. F. Balcan, and L. Song. Scalable kernel methods via doubly stochastic gradients. In *Advances in Neural Information Processing Systems 27*, pages 3041–3049. Curran Associates, Inc., 2014.

[16] Lisandro D. Dalcin, Rodrigo R. Paz, Pablo A. Kler, and Alejandro Cosimo. Parallel distributed computing using python. *Advances in Water Resources*, 34(9):1124 – 1139, 2011. New Computational Methods and Software Tools.

[17] S. H. Dau, W. Song, Z. Dong, and C. Yuen. Balanced sparsest generator matrices for mds codes. In *2013 IEEE International Symposium on Information Theory*, pages 1889–1893, July 2013.

[18] Jeffrey Dean, Greg Corrado, Rajat Monga, Kai Chen, Matthieu Devin, Mark Mao, Marc'Aurelio Ranzato, Andrew Senior, Paul Tucker, Ke Yang, Quoc V. Le, and Andrew Y. Ng. Large scale distributed deep networks. In *Advances in Neural Information Processing Systems 25*. NIPS 2012, 2012.

[19] A. Defazio and T. Caetano. A convex formulation for learning scale-free networks via submodular relaxation. In *Advances in Neural Information Processing Systems 24*, 2012.

[20] Olivier Devolder, François Glineur, and Yurii Nesterov. First-order methods of smooth convex optimization with inexact oracle. *Mathematical Programming*, 146(1-2):37–75, 2014.

[21] Sanghamitra Dutta, Viveck Cadambe, and Pulkit Grover. Short-dot: Computing large linear transforms distributedly using coded short dot products. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2100–2108. Curran Associates, Inc., 2016.

[22] M. Eberts and I. Steinwart. Optimal regression rates for SVMs using Gaussian kernels. *Electron. J. Statist.*, 7:1–42, 2013.

[23] M. Eberts and I. Steinwart. Optimal Learning Rates for Localized SVMs. *ArXiv e-prints*, July 2015.

[24] Q. Gu and J. Han. Clustered support vector machines. In *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, pages 307–315, 2013.

[25] R. Hable. Universal consistency of localized versions of regularized kernel methods. *J. Mach. Learn. Res.*, 14(1):153–186, January 2013.

[26] A. Hero and B. Rajaratnam. Hub discovery in partial correlation graphical models. *IEEE Trans. on Information Theory*, 58(9):6064–6078, 2012.

[27] Qirong Ho, James Cipar, Henggang Cui, Seunghak Lee, Jin Kyu Kim, Phillip B Gibbons, Garth A Gibson, Greg Ganger, and Eric P Xing.

More effective distributed ml via a stale synchronous parallel parameter server. In *Advances in neural information processing systems*, 2013.

[28] C. J. Hsieh, S. Si, and I. S. Dhillon. A divide-and-conquer solver for kernel support vector machines. In *International Conference on Machine Learning (ICML)*, June 2014.

[29] D. Hsu, S. Kakade, and T. Zhang. Random design analysis of ridge regression. In *COLT*, pages 9.1–9.24, 2012.

[30] C. Ji and L. Seymour. A consistent model selection procedure for markov random fields based on penalized pseudolikelihood. *Ann. Appl. Probab.*, 6(2):423–443, 1996.

[31] Kangwook Lee, Maximilian Lam, Ramtin Pedarsani, Dimitris S. Papailiopoulos, and Kannan Ramchandran. Speeding up distributed machine learning using codes. *CoRR*, abs/1512.02673, 2015.

[32] Mu Li, David G Andersen, Alex J Smola, and Kai Yu. Communication efficient distributed machine learning with the parameter server. In *Advances in Neural Information Processing Systems*, pages 19–27, 2014.

[33] S. Li, M. A. Maddah-Ali, and A. S. Avestimehr. Coded mapreduce. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 964–971, Sept 2015.

[34] Songze Li, Mohammad Ali Maddah-Ali, and Amir Salman Avestimehr. A unified coding framework for distributed computing with straggling servers. *CoRR*, abs/1609.01690, 2016.

[35] Songze Li, Mohammad Ali Maddah-Ali, Qian Yu, and Amir Salman Avestimehr. A fundamental tradeoff between computation and communication in distributed computing. *CoRR*, abs/1604.07086, 2016.

[36] M. Lichman. UCI machine learning repository, 2013.

[37] Han Liu, Kathryn Roeder, and Larry A. Wasserman. Stability approach to regularization selection (stars) for high dimensional graphical models. In *NIPS*, pages 1432–1440, 2010.

[38] Q. Liu and A. T. Ihler. Learning scale free networks by reweighted l1 regularization. *Journal of Machine Learning Research - Proceedings Track*, 15:40–48, 2011.

[39] Qiang Liu and Alexander Ihler. Distributed parameter estimation via pseudo-likelihood. In *International Conference on Machine Learning (ICML)*, pages 1487–1494. JMLR.org, July 2012.

[40] C. McDiarmid. On the method of bounded differences. In *Surveys in Combinatorics*, number 141 in London Mathematical Society Lecture Note Series, pages 148–188. Cambridge University Press, August 1989.

[41] Ioannis Mitliagkas, Ce Zhang, Stefan Hadjis, and Christopher Ré. Asynchrony begets momentum, with an application to deep learning. *CoRR*, abs/1605.09774, 2016.

[42] Shravan Narayanamurthy, Markus Weimer, Dhruv Mahajan, Tyson Condie, Sundararajan Sellamanickam, and S. Sathiya Keerthi. Towards resource-elastic machine learning. *NIPS 2013 BigLearn Workshop*, 2013.

[43] J. Peng, P. Wang, N. Zhou, and J. Zhu. Partial correlation estimation by joint sparse regression models. *JASA*, 104(486):735–746, 2009.

[44] D. Politis, J.P. Romano, and M. Wolf. *Subsampling*. Springer series in statistics. Springer Verlag, 1999.

[45] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems 20*, pages 1177–1184, 2007.

[46] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomization in learning. In *Advances in Neural Information Processing Systems 21*, pages 1313–1320, 2008.

[47] G. Raskutti, M. J. Wainwright, and B. Yu. Minimax-optimal rates for sparse additive models over kernel classes via convex programming. *J. Mach. Learn. Res.*, 13:389–427, February 2012.

[48] G. Raskutti, M. J. Wainwright, and B. Yu. Early stopping and non-parametric regression: An optimal data-dependent stopping rule. *J. Mach. Learn. Res.*, 15(1):335–366, January 2014.

[49] P. Ravikumar, M. J. Wainwright, and J. Lafferty. High-dimensional ising model selection using ℓ_1-regularized logistic regression. *Annals of Statistics*, 38(3):1287–1319, 2010.

[50] L. Rosasco, M. Belkin, and E. De Vito. On learning with integral operators. *J. Mach. Learn. Res.*, 11:905–934, March 2010.

[51] A. Rudi, R. Camoriano, and L. Rosasco. Less is more: Nyström computational regularization. In *Advances in Neural Information Processing Systems 28*, pages 1648–1656. Curran Associates, Inc., 2015.

[52] N. Segata and E. Blanzieri. Fast and scalable local kernel machines. *JMLR*, 11:1883–1926, Jun 2010.

[53] Robert J. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley-Interscience, 1981.

[54] S. Smale and D.-X. Zhou. Learning theory estimates via integral operators and their approximations. *Constructive Approximation*, 26(2):153–172, 2007.

[55] I. Steinwart and A. Christmann. *Support Vector Machines*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[56] I. Steinwart, D. R. Hush, and C. Scovel. Optimal rates for regularized least squares regression. In *COLT*, 2009.

[57] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis. Gradient Coding. *ArXiv e-prints*, December 2016.

[58] R. Tandon and P. Ravikumar. On the difficulty of learning power law graphical models. In *In IEEE International Symposium on Information Theory (ISIT)*, 2013.

[59] R. Tandon, S. Si, P. Ravikumar, and I. Dhillon. Kernel Ridge Regression via Partitioning. *ArXiv e-prints*, aug 2016.

[60] Rashish Tandon and Pradeep Ravikumar. Learning graphs with a few hubs. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, pages I–602–I–610. JMLR.org, 2014.

[61] E. Yang and P. Ravikumar. On the use of variational inference for learning discrete graphical models. In *International Conference on Machine learning (ICML)*, 28, 2011.

[62] Y. Yang, M. Mert Pilanci, and M. J. Wainwright. Randomized sketches for kernels: Fast and optimal non-parametric regression. *CoRR*, abs/1501.06195, 2015.

[63] I. E.-H. Yen, T.-W. Lin, S.-D. Lin, P. Ravikumar, and I. S. Dhillon. Sparse random feature algorithm as coordinate descent in hilbert space.

In *Advances in Neural Information Processing Systems 27*, pages 2456–2464. Curran Associates, Inc., 2014.

[64] Matei Zaharia, Andy Konwinski, Anthony D Joseph, Randy H Katz, and Ion Stoica. Improving mapreduce performance in heterogeneous environments. In *OSDI*, volume 8, page 7, 2008.

[65] H. Zhang, A. C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2126–2136, 2006.

[66] T. Zhang. Learning bounds for kernel regression using effective data dimensionality. *Neural Computation*, 17:2077–2098, 2005.

[67] Y. Zhang, J. Duchi, and M. J. Wainwright. Divide and conquer kernel ridge regression. In *COLT*, pages 592–617, 2013.

Vita

Rashish Tandon received a Bachelor of Technology (B.Tech) degree, and a Master of Technology (M.Tech) degree in Computer Science from the Indian Institute of Technology (IIT), Kanpur in 2011. His research interests span Machine Learning and Optimization in high-dimensional, big data and distributed settings.

Permanent address: rashish.tandon@gmail.com

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.